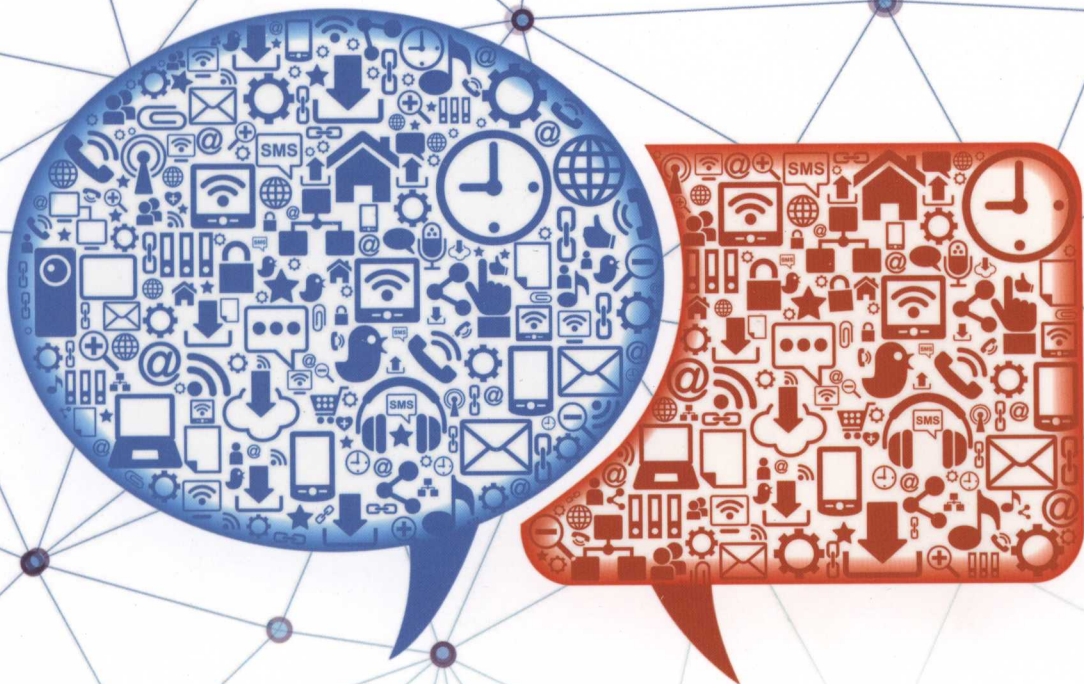


版权注意事项：

- 1、书籍版权归作者和出版社所有
- 2、本PDF仅限用于个人获取知识，进行私底下的知识交流
- 3、PDF获得者不得在互联网上以任何目的进行传播
- 4、如觉得书籍内容很赞，请购买正版实体书，支持作者
- 5、请于下载PDF后24小时内删除本PDF。



图文代码快速理解小程序基本原理和应用方法

海量案例，边练边学

综合实战，感受真实商业项目制作过程

扫二维码看高清精讲视频

微信小程序开发

图解案例教程

附精讲视频

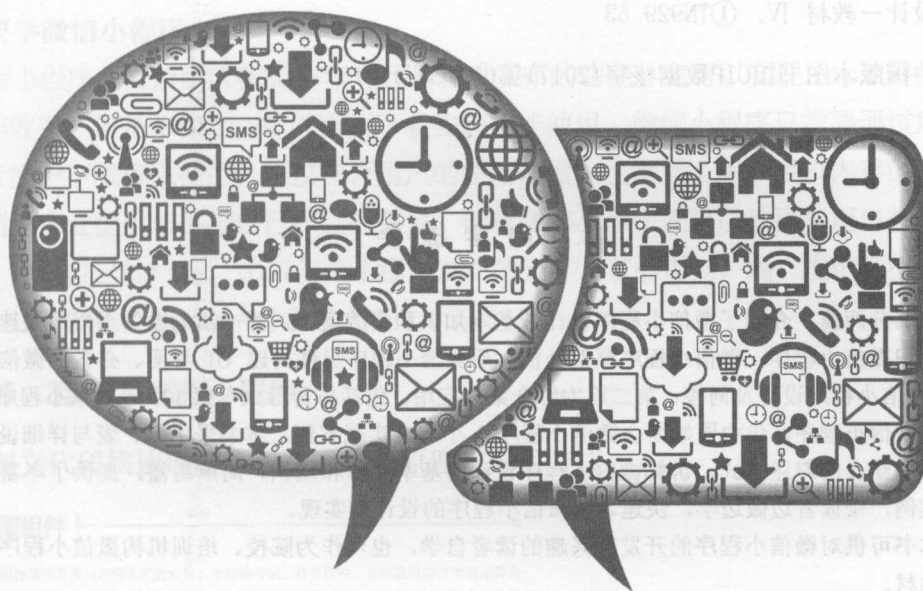
北风网合作出品 刘刚 | 著



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS



微信小程序开发

图解案例教程

附精讲视频

北风网合作出品 刘刚 | 著

人民邮电出版社

北京

图书在版编目 (CIP) 数据

微信小程序开发图解案例教程：附精讲视频 / 刘刚
著. — 北京：人民邮电出版社，2017.5
ISBN 978-7-115-45045-6

I. ①微… II. ①刘… III. ①移动终端—应用程序—
程序设计—教材 IV. ①TN929.53

中国版本图书馆CIP数据核字(2017)第054735号

内 容 提 要

本书分两篇，介绍了微信小程序设计的基本知识和实战案例。第一篇为微信小程序快速入门，包括认识微信小程序、微信小程序框架分析、用微信小程序组件构建 UI 界面、必备的微信小程序 API、微信小程序设计及问答；第二篇为综合案例应用，包括仿智行火车票 12306 微信小程序、仿糗事百科微信小程序、仿中国婚博会微信小程序 3 个综合实战案例。本书采用图、表与详细说明的示例代码相结合的叙述方式，讲解微信小程序设计的基本原理和知识，简单易懂，提供了丰富详尽的实战案例，带读者边做边学，快速掌握微信小程序的设计和实现。

本书可供对微信小程序的开发有兴趣的读者自学，也可作为院校、培训机构微信小程序开发课程的教材。

-
- ◆ 著 刘 刚
责任编辑 桑 珊
责任印制 焦志炜
- ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
三河市海波印务有限公司印刷
- ◆ 开本：787×1092 1/16
印张：22.25 2017 年 5 月第 1 版
字数：554 千字 2017 年 5 月河北第 1 次印刷
-

定价：59.80 元

读者服务热线：(010)81055256 印装质量热线：(010)81055316
反盗版热线：(010)81055315

前言

为什么要学微信小程序

微信小程序是微信团队在2017年1月9日正式发布的功能，它可以实现App软件的原生交互操作效果，但是不像App软件需要下载安装才能使用。微信小程序只需要通过用户扫一扫或者搜一下就可以使用，不仅符合用户的使用习惯，也解放了用户手机内存的占用，同时给创业企业提供了宣传自己产品的渠道。创建微信小程序就可以被更多用户找到自己的产品，宣传自己的产品。2017年是小程序发布的元年，它的市场广阔，提供了很多就业的机会。让我们赶快成为一名小程序员吧！

使用本书，3步学会微信小程序

Step1 图文代码快速理解小程序的基本原理和应用方法。

3.6 地图组件

map地图组件用来开发与地图有关的应用，如地图导航、打车软件、京东商城的订单轨迹都会用到地图组件。在地图上可以标记覆盖物以及指定一系列的坐标位置，如京东的仓库和客户的收货地址如图3.53所示。

学什么，用来做什么



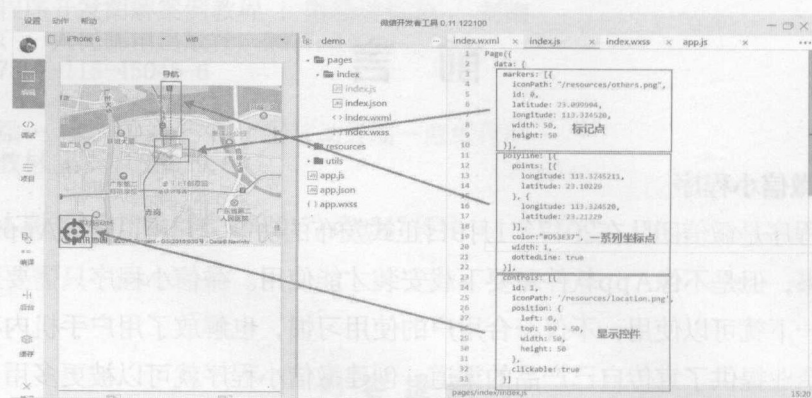
图3.53 京东订单轨迹

map地图组件的具体属性如表3.30所示。

表3.30 map地图的属性

| 属性 | 类型 | 默认值 | 说明 |
|-----------|---------|-----|------------------|
| longitude | Number | | 中心经度 |
| latitude | Number | | 中心纬度 |
| scale | Number | 16 | 缩放级别，取值范围为5~18 |
| markers | Array | | 标记点 |
| covers | Array | | 即将移除，请使用 markers |
| autoplay | Boolean | | 是否自动播放 |
| polyline | Array | | 路线 |

图+表+示例代码，详解基本原理



```
onReady: function (e) {  
    // 使用 wx.createContext 得到画布 context  
    var context = wx.createContext();  
    context.setStrokeStyle("red");  
    context.setLineWidth(5);  
    context.rect(0, 0, 200, 200);  
    context.stroke();  
    context.setStrokeStyle("red");  
    context.setLineWidth(5);  
    context.moveTo(100, 100);  
    context.arc(100, 100, 50, 0, 2 * Math.PI, true);  
    context.moveTo(100, 100);  
    context.arc(100, 100, 50, 0, 2 * Math.PI, false);  
    context.moveTo(100, 100);  
    context.arc(100, 100, 50, 0, 2 * Math.PI, true);  
    context.moveTo(100, 100);  
    context.arc(100, 100, 50, 0, 2 * Math.PI, false);  
    context.stroke();  
    // 调用 wx.drawCanvas 通过 canvasId 指定在哪个画布上绘制，通过 actions 指定绘制动作  
    wx.drawCanvas({  
        canvasId: "firstCanvas",  
        actions: context.getActions(),  
    });  
}
```

创建画布绘图环境

设置矩形位置、大小、边框颜色、线宽

设置线条颜色和线宽

移动画笔坐标位置，绘制弧形

移动画笔坐标位置，绘制弧形

移动画笔坐标位置，绘制弧形

移动画笔坐标位置，绘制弧形

详细代码说明，一看就懂

Step2 沙场大练兵——边做边学。

4.13 沙场大练兵：仿豆瓣电影微信小程序

学完马上实战演练



案例最终效果，学完本章就会做

Step3 综合实战，感受真实商业项目的制作过程。

第6章 综合案例：仿智行火车票12306 微信小程序

智行火车票是一款收取费用的用来自动查询预订火车票的软件，可以实时监控票数的多少，与铁道部数据实时同步。通过该软件可以直接在12306上订票，可以查询、预定和购买。它还对12306购票流程进行了大量优化，使用户购票更加快捷。软件还额外提供了智能查询和火车票监控功能。

完整运用
所学知识



不止讲实现
还讲调研和设计

□ 小刚老师简介

本名刘刚，参与过多个软件项目的研发、设计和管理工作，拥有项目管理师高级认证、项目监理师中级认证，出版过的图书有《原型设计大师：Axure RP网站与APP设计从入门到精通》《Axure RP原型设计图解微课视频教程（Web+App）》；曾在中国擎天公司、神州软件子公司任职，在项目管理和项目实践、软件设计等方面经验丰富；曾负责纪检监察廉政监督监管平台产品的设计与开发、国家邮政局项目的设计与开发、政务大数据项目的设计与开发等。

平台支撑，免费赠送资源

- ☑ 全部案例源代码、素材、最终文件
- ☑ 全书电子教案

☑ 北风网部分高清精讲视频教程

附赠资源可登录人邮教育社区 (www.ryjiaoyu.com.cn) 或网盘下载使用。

著 者

2017年2月

第1章

第2章

第3章

第4章

第5章

第6章

第7章

第8章

第9章

第10章

第11章

第12章

第13章

第14章

第15章

第16章

第17章

第18章

第19章

第20章

第21章

第22章

第23章

第24章

第25章

第26章

第27章

第28章

第29章

第30章

第31章

第32章

第33章

第34章

第35章

第36章

第37章

第38章

第39章

第40章

第41章

第42章

第43章

第44章

第一篇

微信小程序快速入门

1

第1章 认识微信小程序 1

1.1 微信小程序介绍 1

1.1.1 初识微信小程序 1

1.1.2 微信小程序的功能 2

1.1.3 微信小程序能否取代App 3

1.1.4 微信小程序的发展历程 3

1.1.5 微信小程序带来的机会 3

1.2 微信小程序开发准备 3

1.2.1 基础技术准备 3

1.2.2 开发准备 4

1.3 微信小程序开发工具的使用 5

1.3.1 创建项目 5

1.3.2 编辑 7

1.3.3 常用快捷键 9

1.3.4 调试 10

1.3.5 项目 12

1.3.6 编译 13

1.3.7 前台/后台 13

1.3.8 缓存 14

1.4 沙场大练兵：Hello World 的
创建 14

1.5 小结 17

2

第2章 微信小程序框架分析 18

2.1 微信小程序目录结构介绍 18

2.1.1 框架全局文件 18

2.1.2 工具类文件 23

2.1.3 框架页面文件 24

2.1.4 小试牛刀：制作猫眼电影
底部标签导航 24

2.2 微信小程序注册程序的应用 27

2.3 微信小程序注册页面的使用 29

2.3.1 页面初始化数据 30

2.3.2 生命周期函数 30

2.3.3 页面相关事件处理函数 30

2.3.4 页面路由管理 31

2.3.5 自定义函数 32

2.3.6 setData设值函数 32

2.4 微信小程序如何绑定数据 33

2.4.1 组件属性绑定 33

2.4.2 控制属性绑定 34

2.4.3 关键字绑定 34

2.4.4 运算 34

2.4.5 小试牛刀：天气微信
小程序 35

2.5 微信小程序条件渲染 38

| | | |
|-------|-----------------------|----|
| 2.5.1 | wx:if 判断单个组件 | 38 |
| 2.5.2 | block wx:if 判断多个组件 | 39 |
| 2.6 | 微信小程序列表渲染 | 39 |
| 2.6.1 | wx:for 列表渲染单个组件 | 39 |
| 2.6.2 | block wx:for 列表渲染多个组件 | 39 |
| 2.6.3 | wx:key 指定唯一标识符 | 40 |
| 2.7 | 微信小程序定义模板 | 40 |
| 2.7.1 | 定义模板 | 40 |
| 2.7.2 | 使用模板 | 41 |
| 2.8 | 微信小程序的引用功能 | 41 |
| 2.8.1 | import 引用 | 41 |
| 2.8.2 | include 引用 | 42 |
| 2.9 | 沙场大练兵：仿香哈菜谱微信小程序 | 42 |
| 2.9.1 | 底部标签导航设计 | 43 |
| 2.9.2 | 宫格导航设计 | 45 |
| 2.9.3 | 香哈头条初始化数据 | 48 |
| 2.9.4 | 香哈头条列表渲染及绑定数据 | 50 |
| 2.9.5 | 香哈头条模板的引用 | 54 |
| 2.10 | 小结 | 56 |

第3章 用微信小程序组件构建UI界面 57

| | | |
|-------|--------------------|----|
| 3.1 | 视图容器组件 | 57 |
| 3.1.1 | view视图容器 | 57 |
| 3.1.2 | scroll-view可滚动视图区域 | 58 |
| 3.1.3 | swiper滑块视图容器 | 61 |

| | | |
|--------|-------------------------|-----|
| 3.2 | 基础内容组件 | 65 |
| 3.2.1 | icon图标组件 | 65 |
| 3.2.2 | text文本组件 | 66 |
| 3.2.3 | progress进度条组件 | 67 |
| 3.3 | 丰富的表单组件 | 68 |
| 3.3.1 | button按钮 | 68 |
| 3.3.2 | checkbox多项选择器 | 70 |
| 3.3.3 | radio单项选择器 | 71 |
| 3.3.4 | input单行输入框 | 72 |
| 3.3.5 | textarea多行输入框 | 75 |
| 3.3.6 | label组件 | 77 |
| 3.3.7 | picker滚动选择器 | 79 |
| 3.3.8 | slider滑动选择器 | 85 |
| 3.3.9 | switch开关选择器 | 87 |
| 3.3.10 | form表单 | 89 |
| 3.4 | 导航组件 | 92 |
| 3.4.1 | navigator页面链接组件 | 92 |
| 3.4.2 | wx.navigateTo保留当前页跳转 | 94 |
| 3.4.3 | wx.redirectTo关闭当前页跳转 | 95 |
| 3.4.4 | wx.switchTab跳转到tabBar页面 | 96 |
| 3.4.5 | wx.navigateBack返回上一页 | 97 |
| 3.4.6 | 设置导航条 | 98 |
| 3.5 | 媒体组件 | 100 |
| 3.5.1 | audio音频 | 100 |
| 3.5.2 | image图片 | 103 |
| 3.5.3 | video视频 | 107 |
| 3.6 | 地图组件 | 110 |
| 3.7 | 画布组件 | 114 |

3.8 沙场大练兵：表单登录注册微信小程序 116

3.8.1 登录设计 117

3.8.2 手机号注册设计 123

3.8.3 企业用户注册设计 128

3.9 小结 136

第4章 必备的微信小程序API 137

4.1 请求服务器数据API 137

4.2 文件上传与下载API 140

4.2.1 wx.uploadFile文件上传 140

4.2.2 wx.downloadFile文件下载 143

4.3 WebSocket会话API 145

4.4 图片处理API 150

4.4.1 wx.chooseImage (OBJECT)选择图片 150

4.4.2 wx.previewImage (OBJECT)预览图片 151

4.4.3 x.getImageInfo(OBJECT)获得图片信息 152

4.5 文件操作API 153

4.5.1 wx.saveFile保存文件到本地 153

4.5.2 wx.getSavedFileList获取本地文件列表 154

4.5.3 wx.getSavedFileInfo获取本地文件信息 156

4.5.4 wx.removeSavedFile删除本地文件 156

4.5.5 wx.openDocument打开文档 157

4.6 数据缓存API 158

4.6.1 数据缓存到本地 158

4.6.2 获取本地缓存数据 160

4.6.3 移除和清理本地缓存数据 164

4.7 位置信息API 165

4.7.1 获得位置、选择位置、打开位置 165

4.7.2 地图组件控制 169

4.8 设备应用API 170

4.8.1 获得系统信息 171

4.8.2 获取网络状态 172

4.8.3 重力感应 172

4.8.4 罗盘 173

4.8.5 拨打电话 173

4.8.6 扫码 174

4.9 交互反馈API 174

4.9.1 消息提示框 174

4.9.2 模态弹窗 176

4.9.3 操作菜单 177

4.10 登录API 179

4.11 微信支付API 183

4.12 分享API 184

4.13 沙场大练兵：仿豆瓣电影微信小程序 185

4.13.1 电影顶部页签切换效果 186

4.13.2 电影海报轮播效果 190

4.13.3 电影列表方式布局 192

4.13.4 电影详情页布局 197

| | | |
|--------|---------|-----|
| 4.13.5 | 项目上传与预览 | 206 |
| 4.14 | 小结 | 207 |

5

第5章 微信小程序设计及问答 208

| | | |
|-------|------------|-----|
| 5.1 | 微信小程序设计 | 208 |
| 5.1.1 | 突出重点，减少干扰项 | 208 |
| 5.1.2 | 主次动作区分明显 | 208 |
| 5.1.3 | 流程明确，避免打断 | 209 |

第二篇

综合案例应用

6

第6章 综合案例：仿智行火车票12306微信小程序 216

| | | |
|-------|------------|-----|
| 6.1 | 需求描述 | 217 |
| 6.2 | 设计思路及相关知识点 | 219 |
| 6.2.1 | 设计思路 | 219 |
| 6.2.2 | 相关知识点 | 220 |
| 6.3 | 准备工作 | 220 |
| 6.4 | 设计流程 | 221 |
| 6.4.1 | 底部标签导航设计 | 221 |
| 6.4.2 | 海报轮播效果设计 | 224 |
| 6.4.3 | 火车票查询界面设计 | 226 |
| 6.4.4 | 火车票列表设计 | 238 |
| 6.4.5 | 个人中心界面设计 | 253 |
| 6.4.6 | 抢票界面设计 | 262 |
| 6.4.7 | 项目上传和预览 | 271 |

| | | |
|-------|-----------|-----|
| 5.1.4 | 局部加载反馈 | 210 |
| 5.1.5 | 模态窗口加载反馈 | 210 |
| 5.1.6 | 弹出式操作结果 | 211 |
| 5.1.7 | 模态对话框操作结果 | 212 |
| 5.1.8 | 结果页 | 212 |
| 5.1.9 | 表单填写友好提示 | 212 |
| 5.2 | 微信小程序问答 | 213 |
| 5.3 | 小结 | 215 |

7

第7章 综合案例：仿糗事百科微信小程序 274

| | | |
|-------|--------------|-----|
| 7.1 | 需求描述 | 274 |
| 7.2 | 设计思路及相关知识点 | 276 |
| 7.2.1 | 设计思路 | 276 |
| 7.2.2 | 相关知识点 | 276 |
| 7.3 | 准备工作 | 276 |
| 7.4 | 设计流程 | 277 |
| 7.4.1 | 顶部页签菜单滑动设计 | 277 |
| 7.4.2 | 顶部页签菜单切换效果设计 | 279 |
| 7.4.3 | 糗事列表页设计 | 281 |
| 7.4.4 | 视频列表页设计 | 291 |

| | |
|------------|-----|
| 7.4.5 分享设计 | 294 |
| 7.4.6 项目预览 | 296 |
| 7.5 小结 | 297 |

第8章 综合案例：仿中国婚博会微信小程序 298

| | |
|----------------|-----|
| 8.1 需求描述 | 299 |
| 8.2 设计思路及相关知识点 | 301 |
| 8.2.1 设计思路 | 301 |
| 8.2.2 相关知识点 | 302 |
| 8.3 准备工作 | 302 |

| | |
|---------------------|-----|
| 8.4 设计流程 | 304 |
| 8.4.1 底部标签导航设计 | 304 |
| 8.4.2 海报轮播效果设计 | 307 |
| 8.4.3 宫格导航设计 | 309 |
| 8.4.4 全部分类导航设计 | 313 |
| 8.4.5 现金券下拉菜单筛选条件设计 | 322 |
| 8.4.6 现金券列表页设计 | 325 |
| 8.4.7 婚博会索票界面设计 | 332 |
| 8.4.8 获知渠道弹出层设计 | 337 |
| 8.5 小结 | 344 |

第一篇

微信小程序快速入门

第 1 章 认识微信小程序

2016年1月9日，腾讯公司启动了微信小程序产品的研发，于2017年1月9日正式发布。微信小程序也被称为微信应用号。不同于微信订阅号或公众号，微信小程序被赋予了应用程序的能力，它是一种无需安装即可使用的应用，它实现了应用“触手可及”的梦想，用户扫一扫或者搜一下即可打开应用；也体现了“用完即走”的理念，用户不再需要关心是否安装太多应用的问题。应用将无处不在，随时随地可用，无需卸载。

1.1 微信小程序介绍

1.1.1 初识微信小程序

在微信的“发现”界面中，可以找到小程序的入口，如图1.1所示。小程序的界面和使用方法与App类似，如图1.2所示是几个已发布的常用小程序界面。

微课视频



微信小程序介绍



图1.1 微信小程序入口



图1.2 常用微信小程序界面

提示：若找不到微信小程序入口，可以搜索“小程序示例”来打开小程序入口。

用户需要下载、安装才可以使用App，安装时还会考虑App占用多大存储空间、哪些程序应该卸载掉以释放空间。微信小程序则无需安装，直接使用，不占用存储空间，并在使用微信小程序后，用完即走。例如，我们去餐馆点菜，并不需要去下载这个餐馆的应用程序，只需要在餐馆扫一下二维码，即可在小程序里点赞，之后并不需要去卸载应用程序，直接关闭小程序即可。

微信小程序看起来是程序，但它以完全不同于App的状态出现，具有更灵活的应用组织形态。

1.1.2 微信小程序的功能

小程序提供的功能如下。

1 分享页功能。用户可以将小程序的当前页面分享给好友，如分享北京到上海火车票列表界面，用户打开时是这个页面的实时数据，而不需要再次启动微信小程序。

2 分享对话功能。用户可以将对话分享给好友或者微信群。

3 线下扫码进入微信小程序功能。该功能提示用户附近有哪些微信小程序可以使用，扫描二维码就可以使用微信小程序。

4 挂起状态功能。例如，来电话可以先接电话，接完电话后可以继续使用微信小程序进行相关操作。

5 消息通知功能。商户可以发送消息给接受过服务的用户，用户同时可以使用微信小程序的客服功能联系商户。

小程序不提供的功能如下。

1 小程序没有集中入口，没有应用商店，用户可以通过搜索、扫描二维码、好友分享等多种途径进入微信小程序。

2 小程序没有订阅关系，没有粉丝，只有访问量。

3 小程序不能推送消息。

4 小程序不能做游戏。

1.1.3 微信小程序能否取代App

原生App一般要同时开发iOS和Android两版，而小程序只需要做一版。毫无疑问，这点是小程序最大的优势。从这个角度来看，小程序是“跨平台”的。

在现阶段，开发一套完整逻辑的应用程序，小程序的开发效率是低于App的。小程序独立出了一个封闭的生态。

小程序虽是跨平台的，但是缺乏成熟的组件，缺少统计、绘图组件，以前的echarts和hightcharts都无法使用。

小程序不支持WebView，大量已被静态化好的HTML页面完全没办法在小程序上展示。

小程序想取代Android和iOS还要走很长的路，是蓝海还是死海需要时间来验证。

1.1.4 微信小程序的发展历程

微信小程序从开始研发到正式发布，经历了一年的发展。

1 2016年1月9日，微信团队首次提出应用号的概念。

2 2016年9月22日，微信公众平台对外发送小程序内测邀请，内侧名额200个。

3 2016年11月3日，微信小程序对外公测，开发完成后可以提交审核，但公测期间不能发布。

4 2016年12月28日，张小龙在微信公开课中解答外界对微信小程序的几大疑惑，包括没有应用商店、没有推送消息等。

5 2016年12月30日，微信公众平台对外公告，上线的微信小程序最多可生成10 000个带参数的二维码。

6 2017年1月9日，微信小程序正式上线。

1.1.5 微信小程序带来的机会

微信小程序给很多想做程序员的人员提供了机会，因为它的门槛很低，不需要太难的技术。学习微信小程序开发，就可以成为一名“小程序员”。例如，设计师、学生、创业、待业青年、网虫、策划人员、编辑、草根站长等都可以转做程序员。

1.2 微信小程序开发准备

1.2.1 基础技术准备

微信小程序自定义了一套语言，称为WXML微信标记语言，它的使用方法类似于HTML语言。另外，微信小程序还定义了自己的样式语言WXSS，它兼容了CSS样式，并做了扩展；使用JavaScript来进行业务处理，兼容了大部分JavaScript功能，但仍有一些功能无法使用，所以有一定HTML、CSS、JavaScript技术功底的人学习微信小程序开发会容易很多。

1.2.2 开发准备

1 在<https://mp.weixin.qq.com/>注册微信开发者账号。

微信小程序提供了开发文档（<https://mp.weixin.qq.com/debug/wxadoc/dev/?t=20161107>）。在开发文档里，有“简易教程”“框架”“组件”“API”“工具”以及“Q&A”，如图1.3所示。

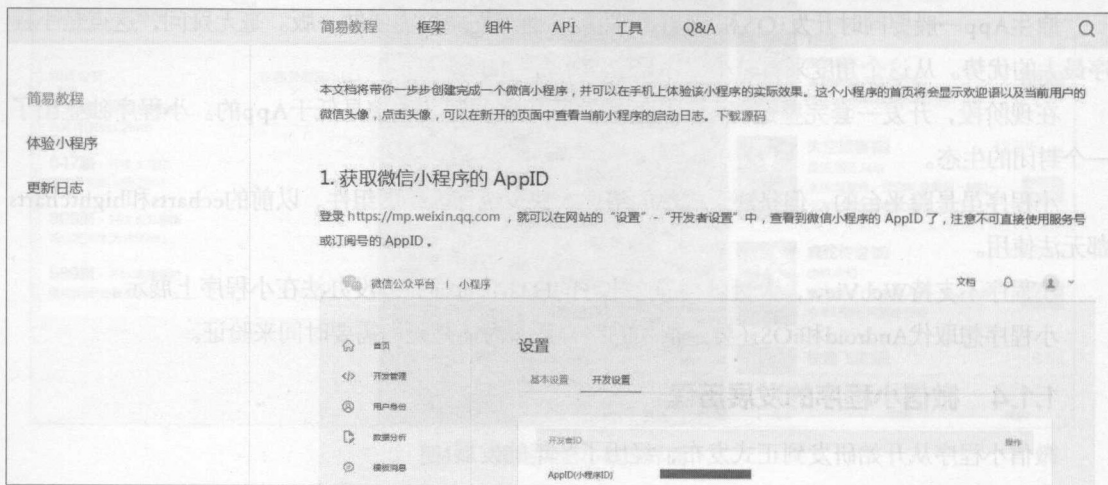


图1.3 开发文档

2 在文档工具里，根据自己的操作系统，下载微信小程序的开发工具，如图1.4所示。

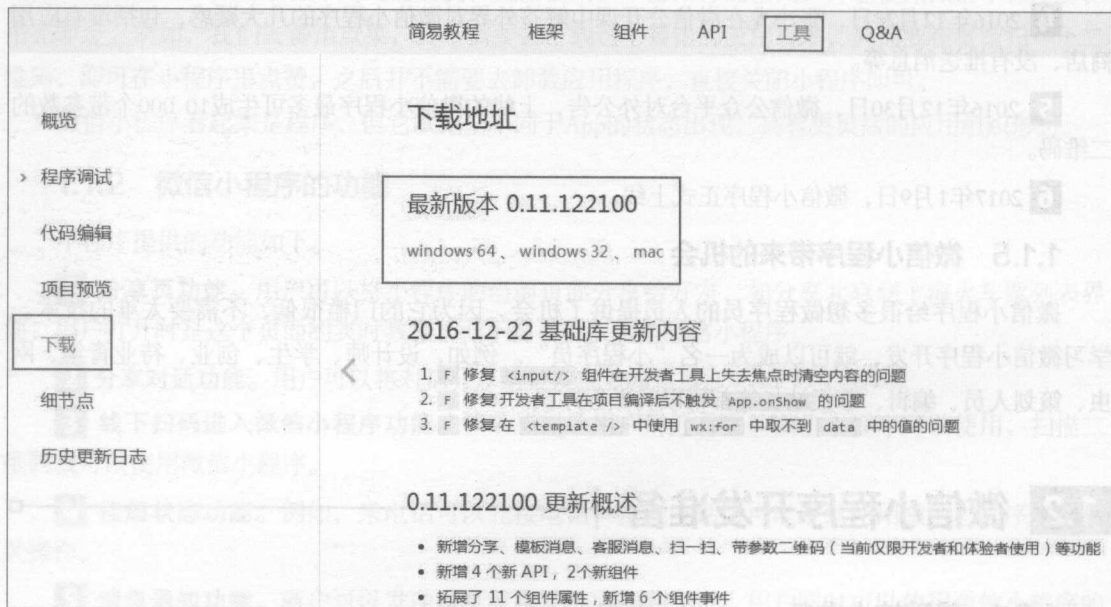


图1.4 下载开发工具

3 按照提示完成开发工具的安裝，安裝完成后，开发工具提供了“本地小程序项目”和“公众号网页开发”两个调试类型，如图1.5所示。

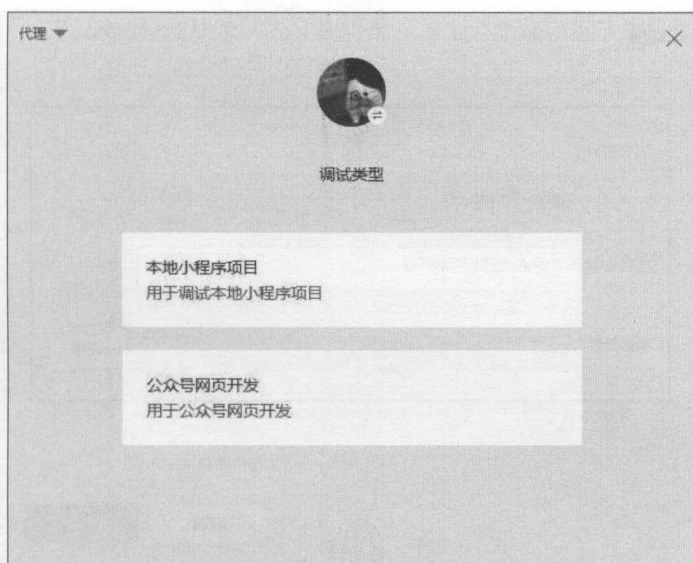


图1.5 开发工具

1.3 微信小程序开发工具的使用

1.3.1 创建项目

在开发工具里单击“本地小程序项目”，进入如图1.6所示界面。

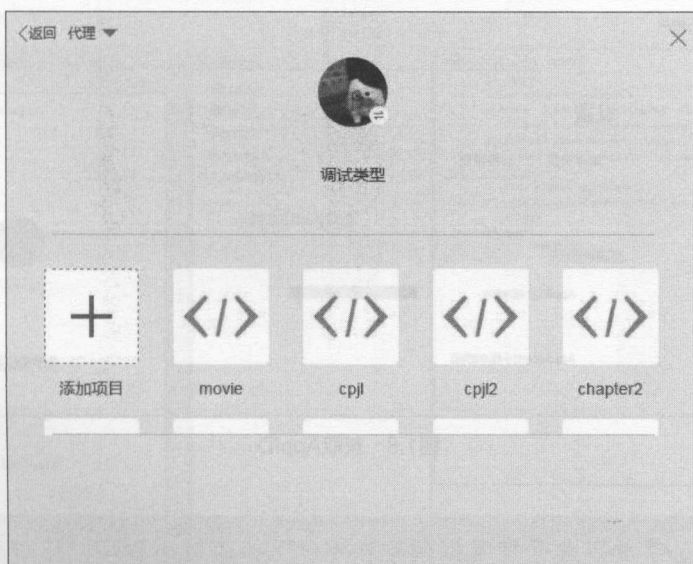
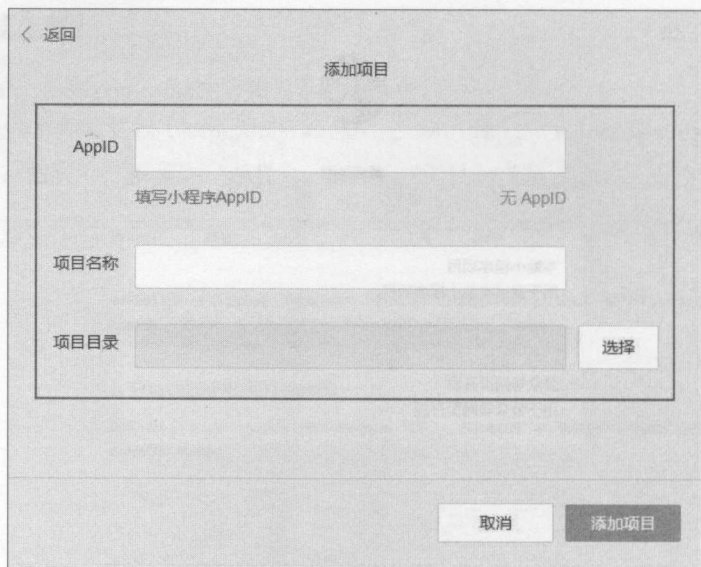


图1.6 添加项目

单击“添加项目”，进入“添加项目”界面，可以添加一个新的项目，在这个界面里需要填写AppID、项目名称、项目目录，如图1.7所示。



获取微信小程序AppID，需要登录 <https://mp.weixin.qq.com>。在网站的“设置”—“开发设置”中，即可查看微信小程序的AppID，如图1.8所示。



注意：不可直接使用服务号或订阅号 AppID；可以不填写AppID，但功能会受限。

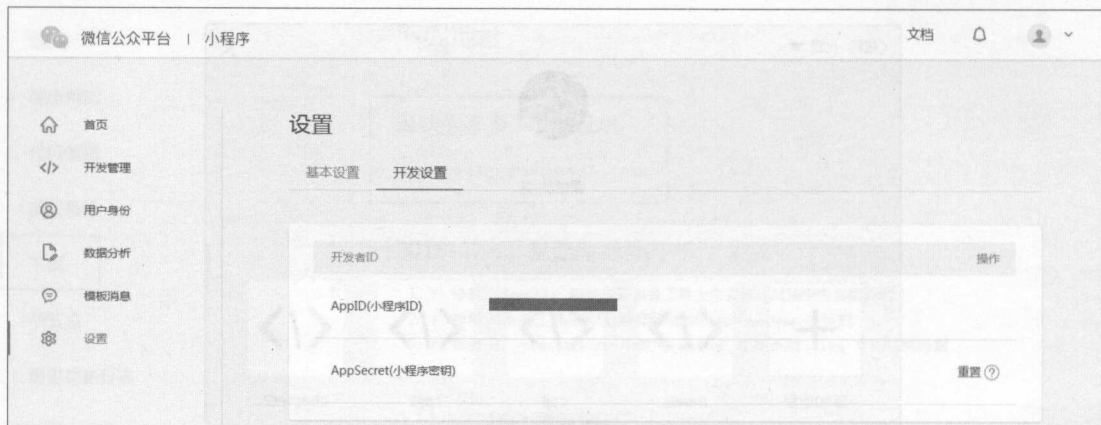


图1.8 获取AppID



注意：如果要以非管理员微信号在手机上体验该小程序，还需要“绑定开发者”，即在“用户身份”—“开发者”模块，绑定需要体验该小程序的微信号。

这里添加一个无AppID的demo项目，在桌面上建立一个“demo”文件夹，勾选“在当前目录中

创建quick start项目”，在demo里创建默认的文件目录，单击“添加项目”即可，如图1.9所示。

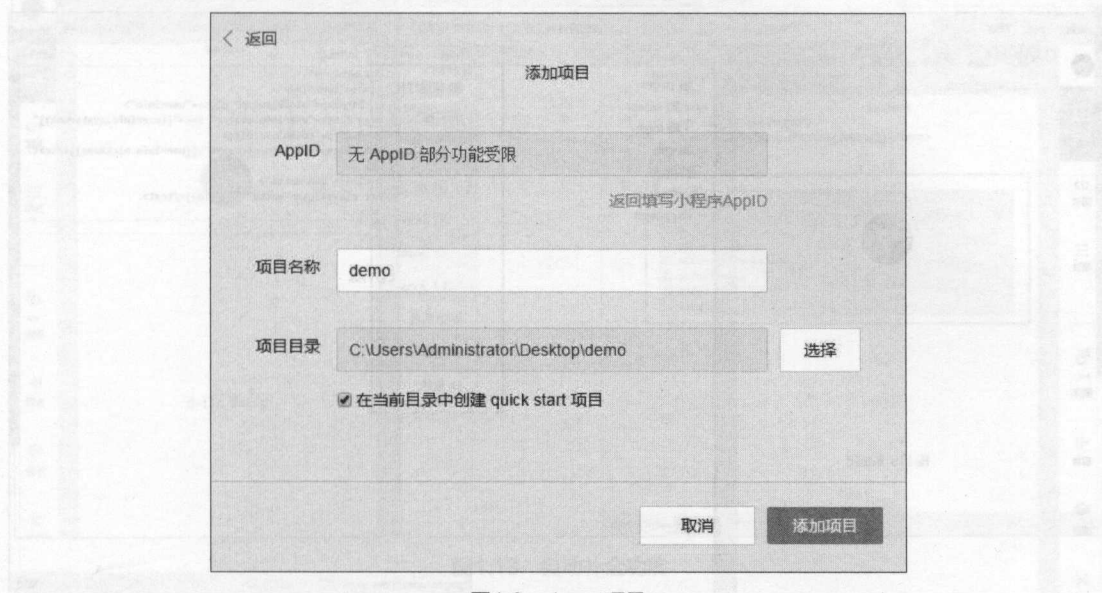


图1.9 demo项目

1.3.2 编辑

进入开发工具，在左侧有7个导航模块：编辑、调试、项目、编译、后台、缓存和关闭。

1 编辑模块用来进行微信小程序的开发，右侧是微信小程序的界面、项目的目录和打开的页面，如图1.10所示。

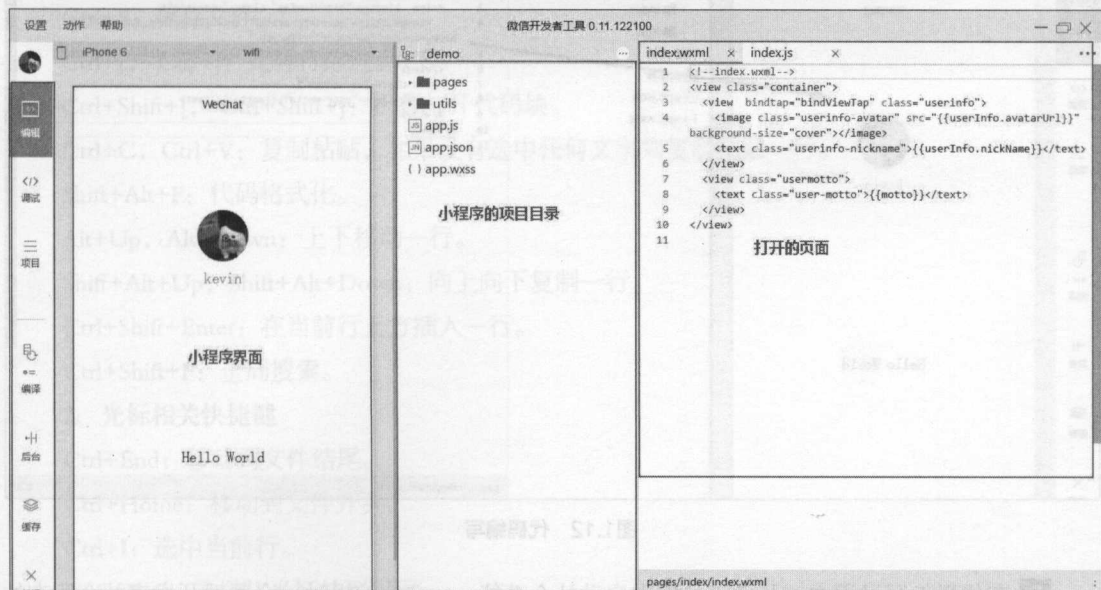


图1.10 编辑功能

2 在硬盘中打开文件的目录，可以新建4种文件：js、json、wxml、wxss，还可对文件进行重命

名、删除和查找，如图1.11所示。

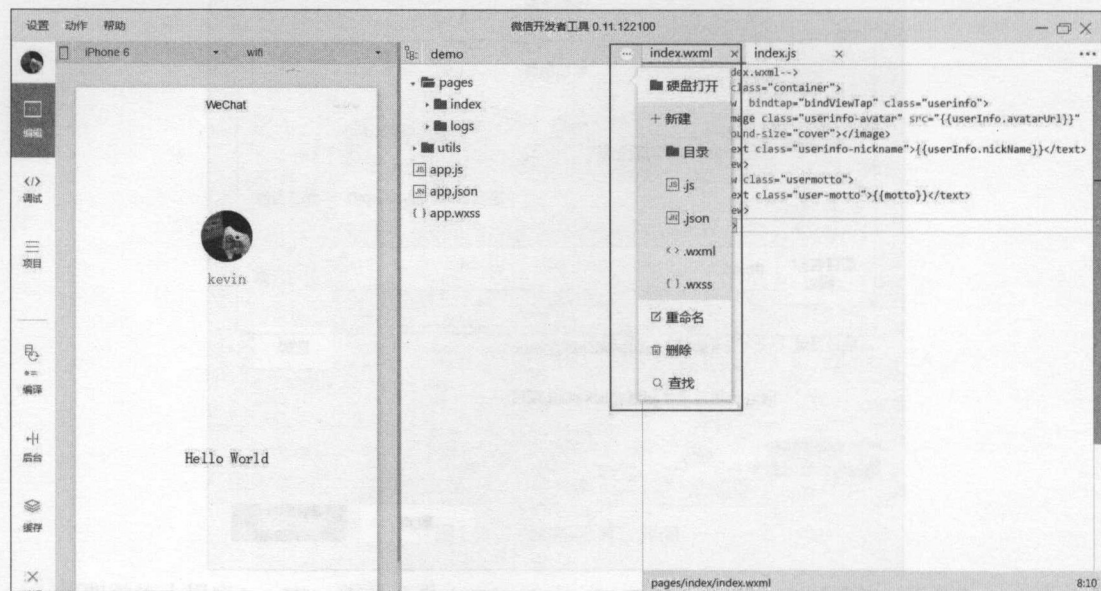


图1.11 文件操作

3 可以通过编辑区左边的模拟器，实时预览编辑的情况。修改 wxss、wxml 文件，会刷新当前 page；修改js文件或者json文件，会重新编译小程序，如图1.12所示。

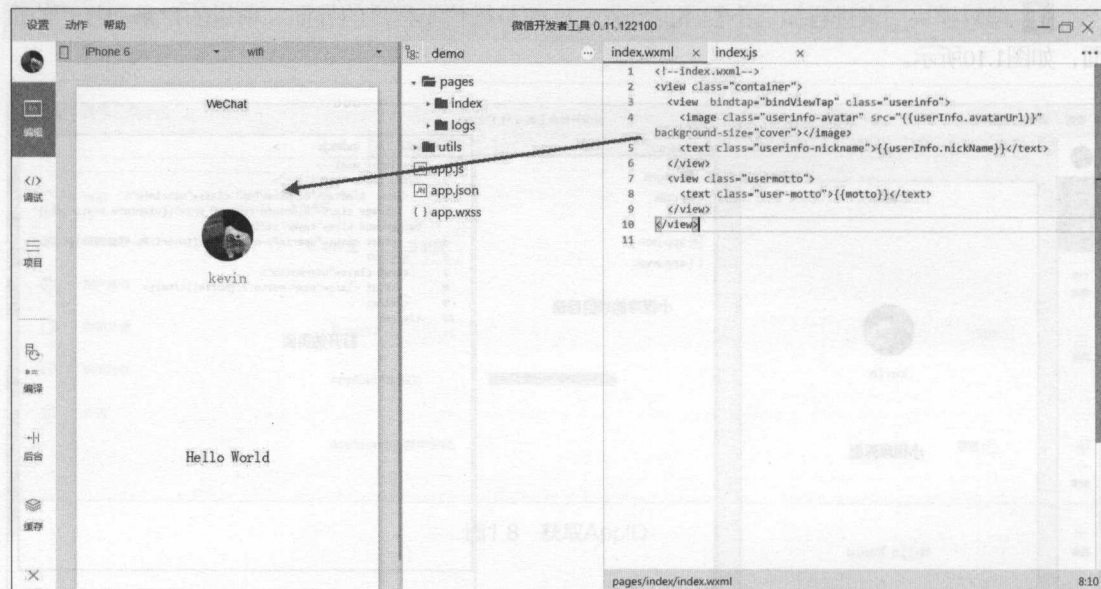


图1.12 代码编写

4 在代码编写过程中，开发工具提供自动补全功能。js 文件编辑时，会帮助开发者补全所有的 API，并给出相关的注释解释；wxml 文件编辑时，会帮助开发者直接写出相关的标签；json 文件编辑时，会帮助开发者补全相关的配置，并给出实时的提示，如图1.13所示。

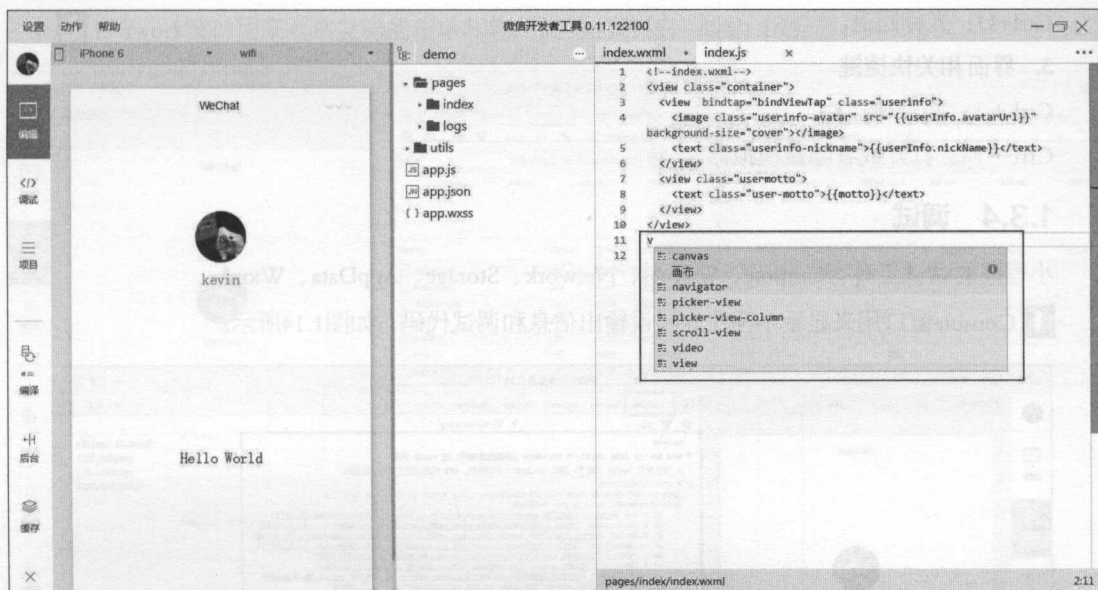


图1.13 自动补全功能

5 开发工具提供自动保存功能。书写代码后，工具会自动帮助用户保存当前代码的编辑状态，直接关闭工具或者切换到别的项目，并不会丢失已经编辑的文件状态。但需要注意的是，只有保存文件，修改内容才会真实地写到硬盘上，并触发实时预览。

1.3.3 常用快捷键

1. 格式调整快捷键

Ctrl+S：保存文件。

Ctrl+[， Ctrl+]：代码行缩进。

Ctrl+Shift+[， Ctrl+Shift+]：折叠打开代码块。

Ctrl+C， Ctrl+V：复制粘贴，如果没有选中任何文字则复制粘贴一行。

Shift+Alt+F：代码格式化。

Alt+Up， Alt+Down：上下移动一行。

Shift+Alt+Up， Shift+Alt+Down：向上向下复制一行。

Ctrl+Shift+Enter：在当前行上方插入一行。

Ctrl+Shift+F：全局搜索。

2. 光标相关快捷键

Ctrl+End：移动到文件结尾。

Ctrl+Home：移动到文件开头。

Ctrl+I：选中当前行。

Shift+End：选择从光标处到行尾。

Shift+Home：选择从行首到光标处。

Ctrl+Shift+L：选中所有匹配。

Ctrl+D：选中匹配。

Ctrl+U：光标后退。

3. 界面相关快捷键

Ctrl + \：隐藏侧边栏。

Ctrl + M：打开或者隐藏模拟器。

1.3.4 调试

小程序的调试工具有Console、Sources、Network、Storage、AppData、Wxml。

1 Console窗口用来显示小程序的错误输出信息和调试代码，如图1.14所示。

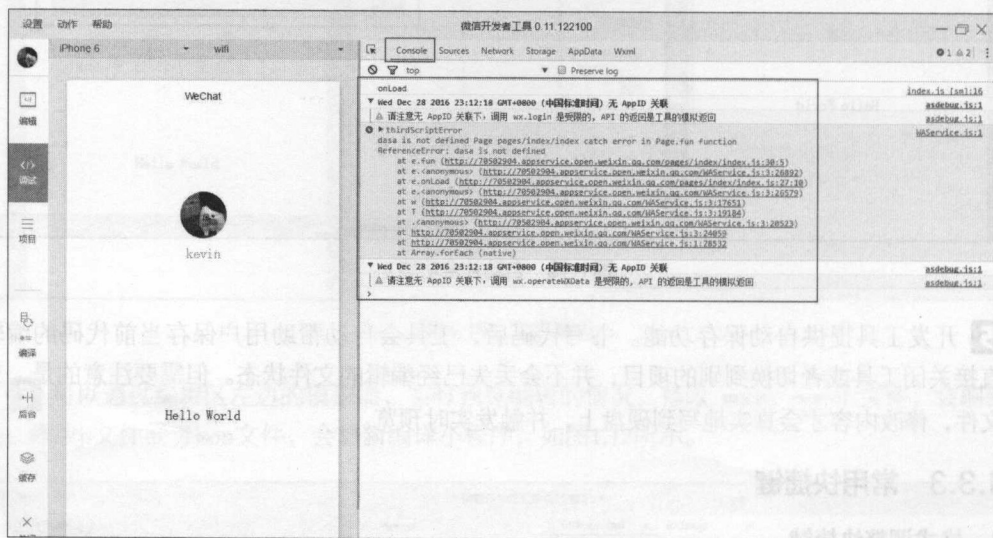


图1.14 Console功能

2 Sources窗口用于显示当前项目的脚本文件，在 Sources中开发者看到的文件是经过处理之后的脚本文件，如图1.15所示。

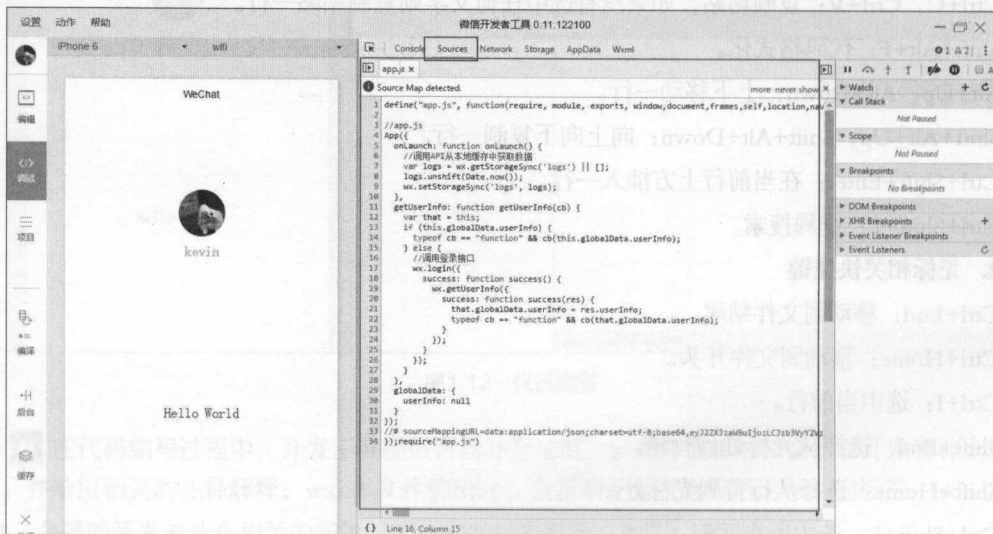


图1.15 Sources功能

3 Network窗口用来观察发送的请求和调用文件的信息，如图1.16所示。

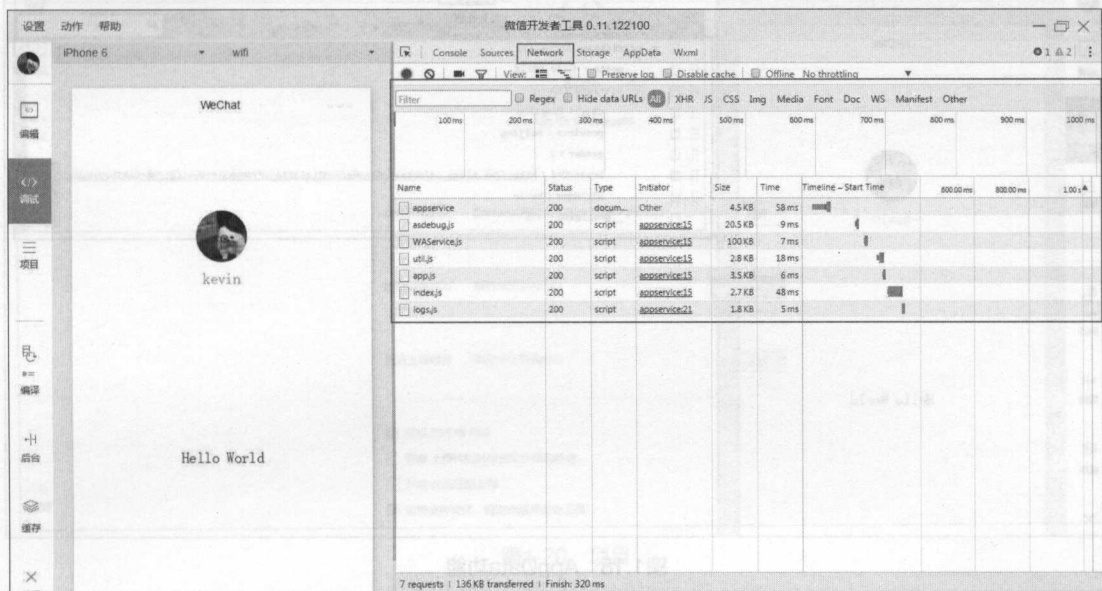


图1.16 Network功能

4 Storage窗口用于显示当前项目使用 `wx.setStorage` 或者 `wx.setStorageSync` 后的数据存储情况，如图1.17所示。

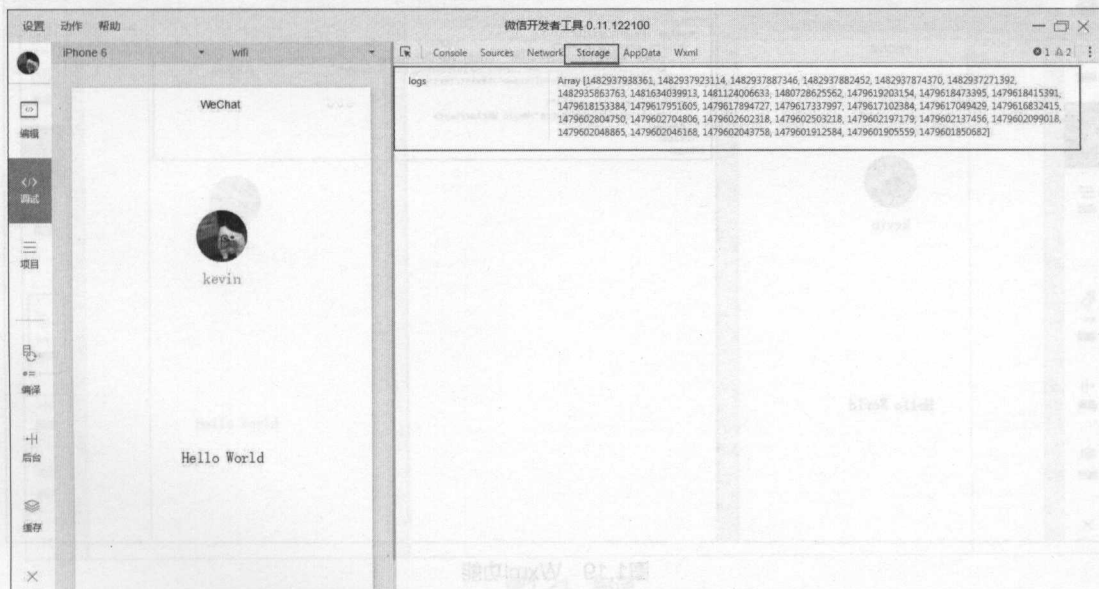


图1.17 Storage功能

5 AppData窗口用于显示当前项目当前时刻的具体数据，实时地反馈项目的数据情况，可以在此处编辑数据，并将其及时地反馈到界面上，如图1.18所示。

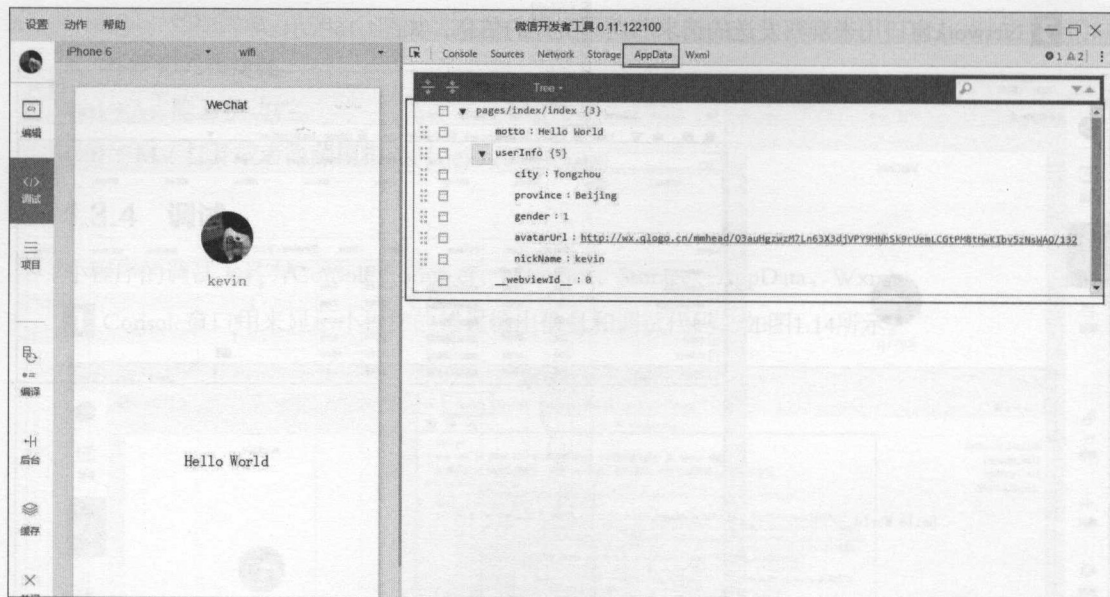


图1.18 AppData功能

6 Wxml窗口用于帮助开发者开发 Wxml 转化后的界面。在这里可以看到真实的页面结构以及结构对应的 wxss 属性，同时可以修改对应的wxss 属性，如图1.19所示。

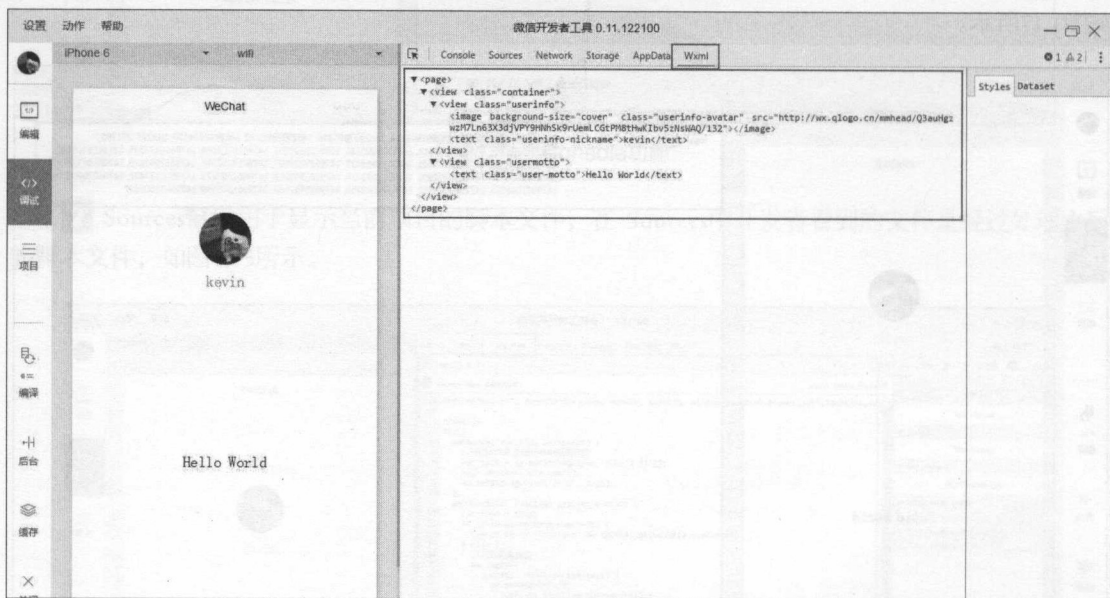


图1.19 Wxml功能

1.3.5 项目

在项目模块里，可以看到微信小程序项目的相关信息，包括项目名称、AppID、项目文件的路径。有AppID的项目，可以在手机上预览微信小程序，如图1.20所示。

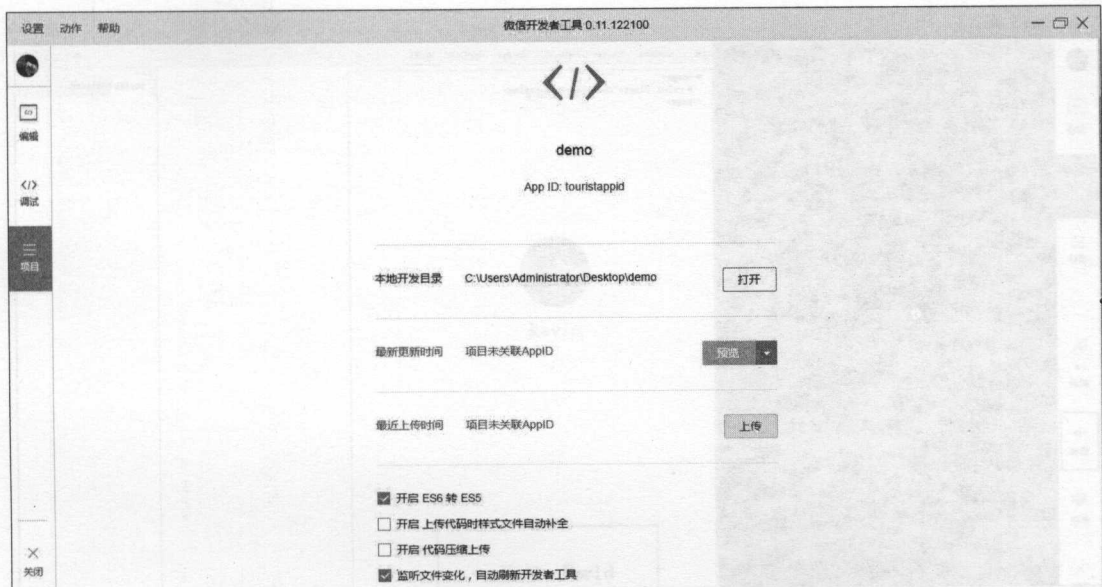


图1.20 项目

1.3.6 编译

编译模块可以对整个项目进行重新编译，如图1.21所示。

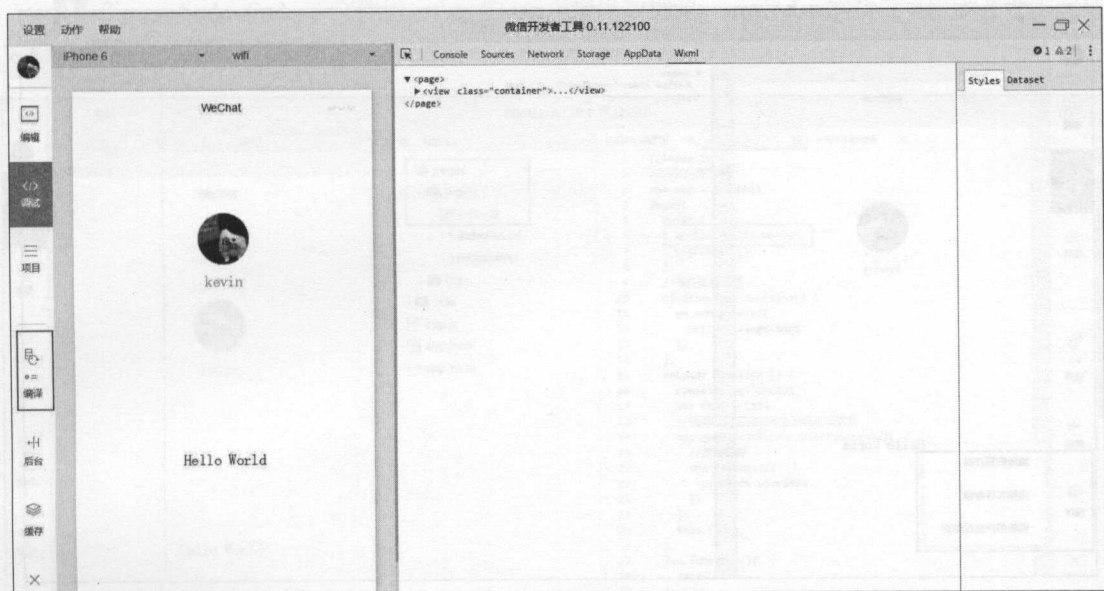


图1.21 编译

1.3.7 前台/后台

后台是指微信小程序从前台进入到后台。例如，在操作微信小程序的过程中，突然来电话，如果接电话，小程序就会从前台进入到后台，重新访问小程序时，又会从后台进入到前台，如图1.22所示。

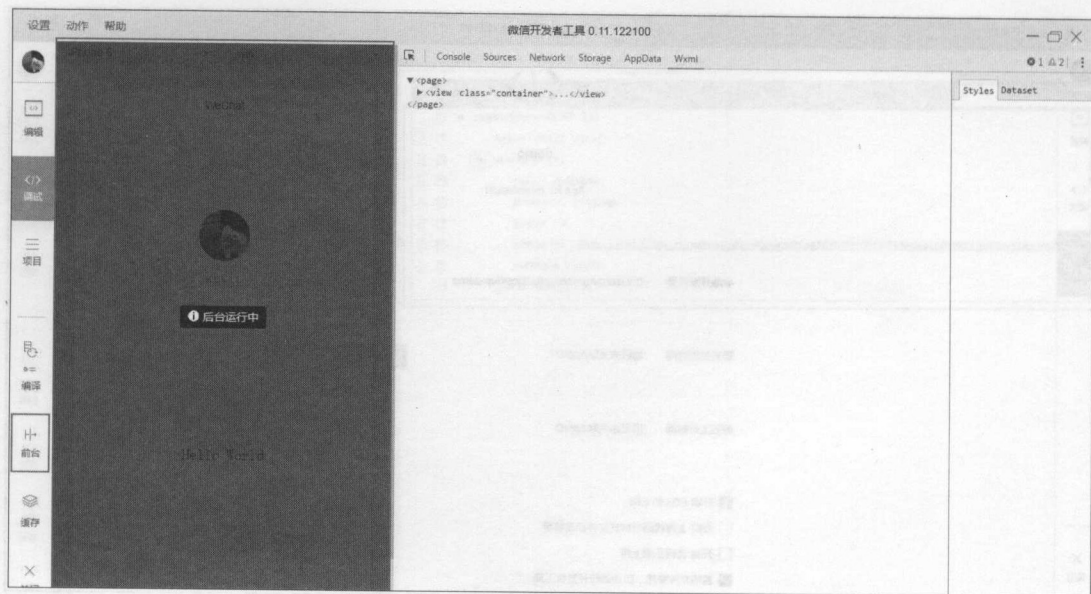


图1.22 前台/后台

1.3.8 缓存

缓存模块用来清除数据存储、文件存储、用户授权数据，如图1.23所示。

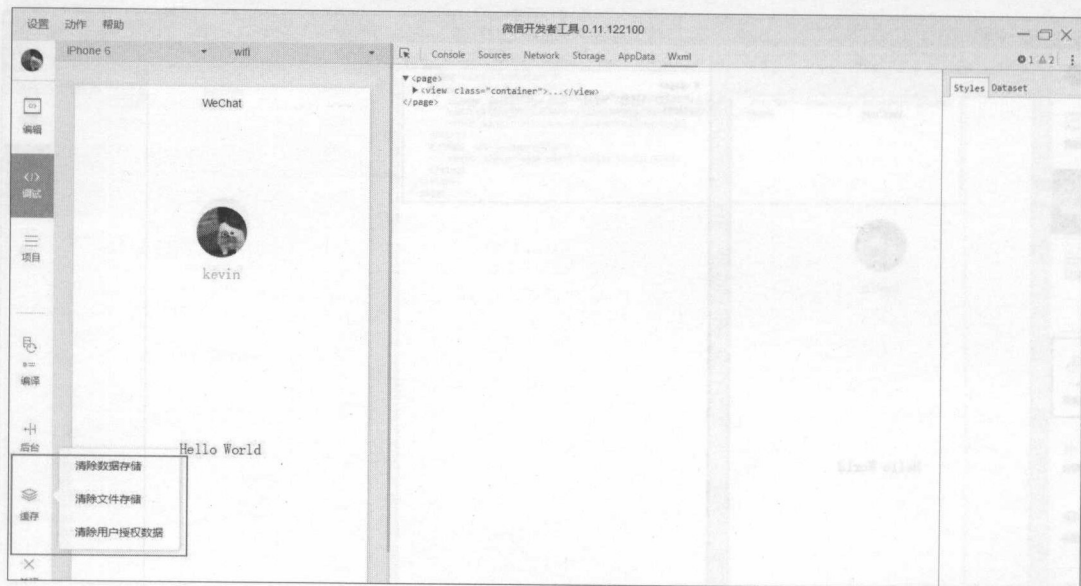


图1.23 缓存

1.4 沙场大练兵：Hello World 的创建

在创建项目之后，开发工具会添加默认的目录和页面，在默认的页面上，可以看到有“Hello

World”文字，如图1.24所示。

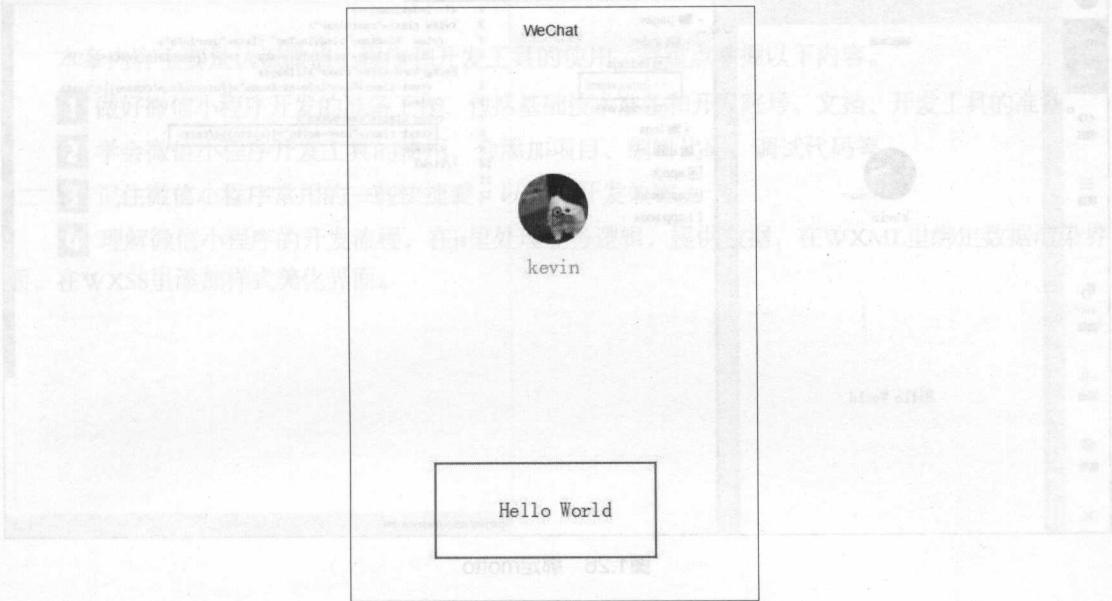


图1.24 Hello World界面

下面，我们分析一下Hello World是怎么创建出来的。

1 在pages/index/index.js文件里，Page的data中提供数据源motto，data的数据可以动态地绑定到WXML页面中，如图1.25所示。

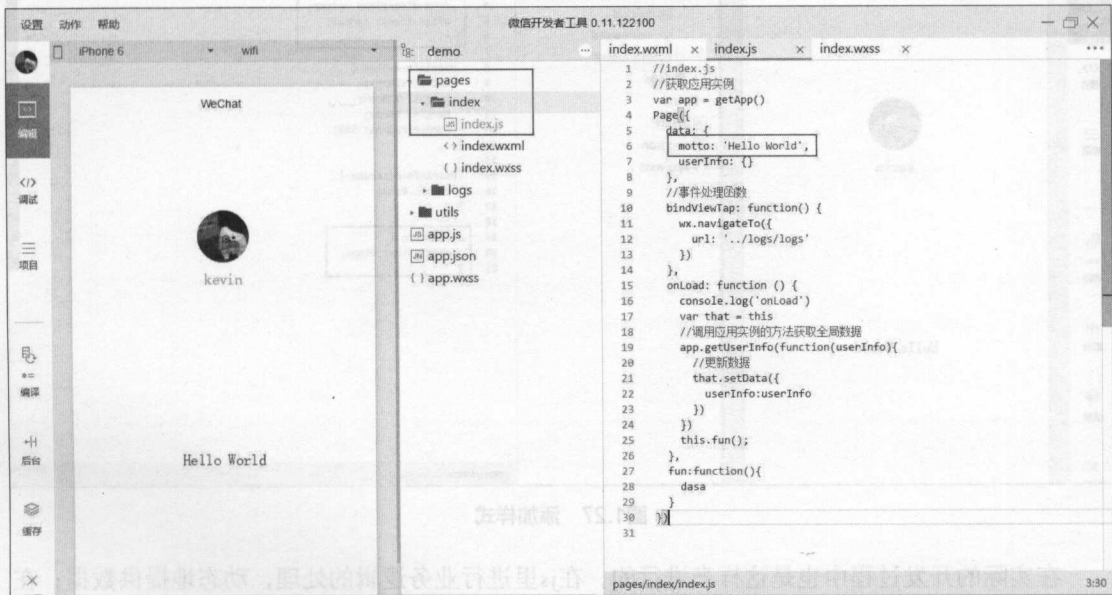


图1.25 motto数据源

2 在pages/index/index.wxml文件里，通过双大括号（{{}}）的方式，将motto绑定到页面中，motto对应的值就可以在页面中显示出来，如图1.26所示。

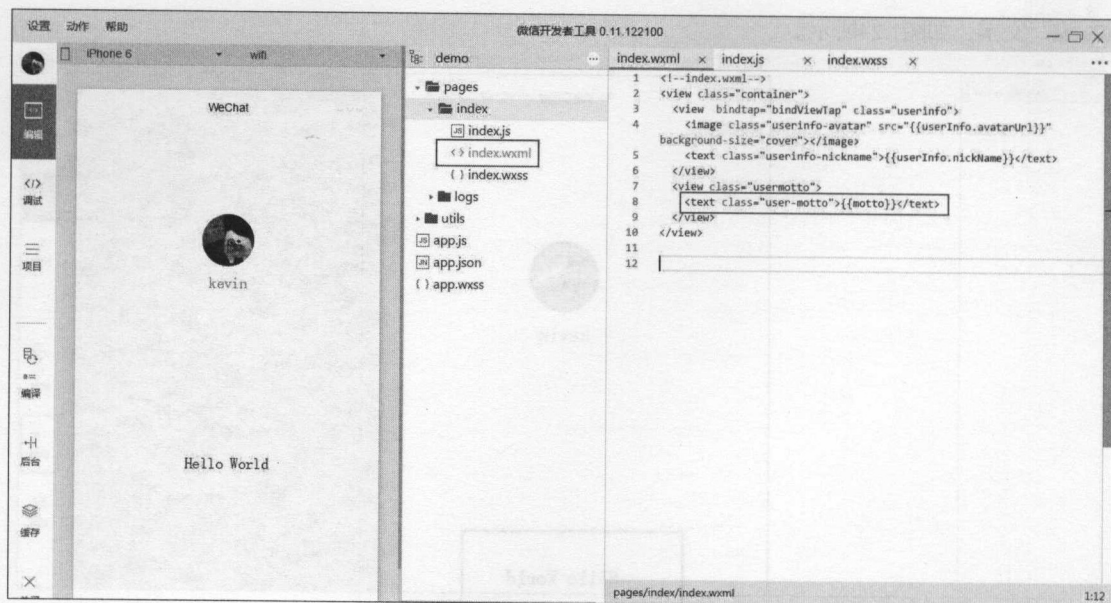


图1.26 绑定motto

3 在pages/index/index.wxss文件里，通过class的方式给Hello World添加样式，距顶部的高度为200px，如图1.27所示。

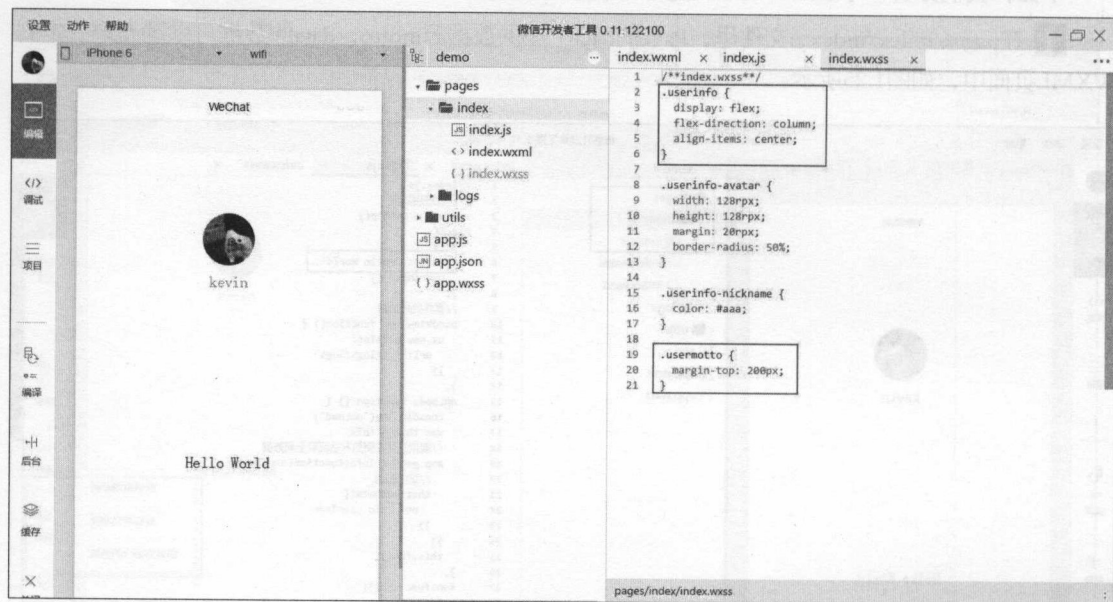


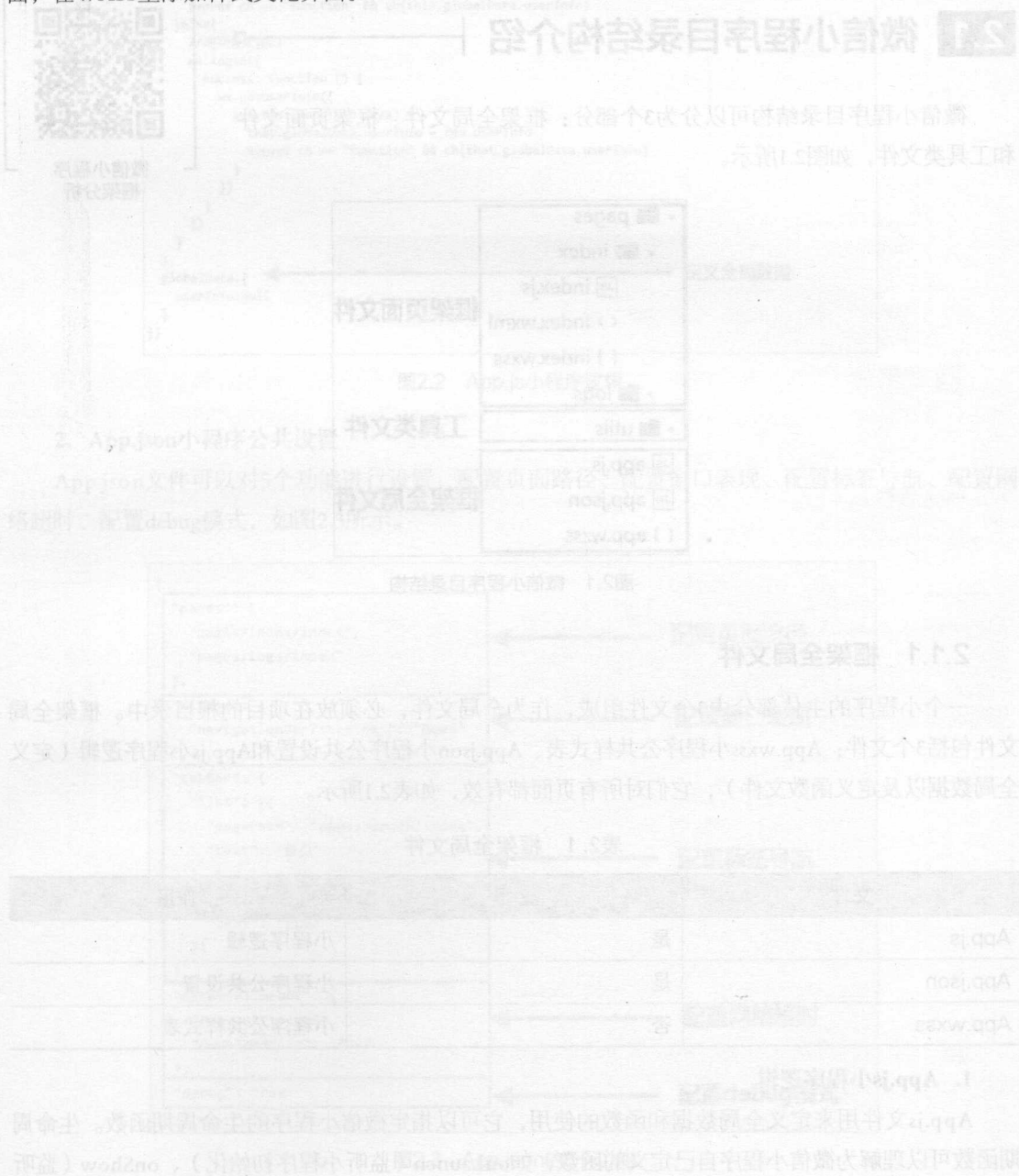
图1.27 添加样式

在实际的开发过程中也是这样来进行的，在js里进行业务逻辑的处理，动态地提供数据；在WXML里绑定数据，渲染界面；在WXSS里添加样式，美化页面。这样，就可以完成微信小程序的开发了。

1.5 小结

本章内容主要是认识微信小程序和开发工具的使用，需重点掌握以下内容。

- 1** 做好微信小程序开发的准备工作，包括基础技术准备和开发账号、文档、开发工具的准备。
- 2** 学会微信小程序开发工具的使用，会添加项目、编辑代码、调试代码等。
- 3** 记住微信小程序常用的一些快捷键，以提高开发效率。
- 4** 理解微信小程序的开发流程，在js里处理业务逻辑，提供数据，在WXML里绑定数据渲染界面，在WXSS里添加样式美化界面。



第2章 微信小程序框架分析

微信小程序框架是进行微信小程序开发必先理解的内容。微信小程序框架让开发者在微信中通过简单、高效的方式开发具有原生App体验的服务。微信小程序框架分为逻辑层和视图层，逻辑层用来处理业务逻辑，而视图层用来渲染页面。视图层描述语言WXML和视图样式WXSS，再加上JavaScript逻辑层语言和json配置文件，构筑起了微信小程序框架。

2.1 微信小程序目录结构介绍

微信小程序目录结构可以分为3个部分：框架全局文件、框架页面文件和工具类文件，如图2.1所示。

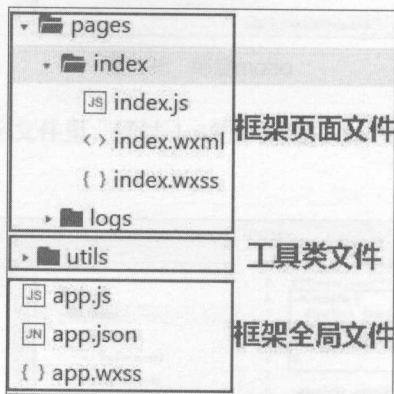


图2.1 微信小程序目录结构

2.1.1 框架全局文件

一个小程序的主体部分由3个文件组成，作为全局文件，必须放在项目的根目录中。框架全局文件包括3个文件：App.wxss小程序公共样式表、App.json小程序公共设置和App.js小程序逻辑（定义全局数据以及定义函数文件），它们对所有页面都有效，如表2.1所示。

表2.1 框架全局文件

| 文件 | 必填 | 作用 |
|----------|----|----------|
| App.js | 是 | 小程序逻辑 |
| App.json | 是 | 小程序公共设置 |
| App.wxss | 否 | 小程序公共样式表 |

1. App.js小程序逻辑

App.js文件用来定义全局数据和函数的使用，它可以指定微信小程序的生命周期函数。生命周期函数可以理解为微信小程序自己定义的函数，如onLaunch（监听小程序初始化）、onShow（监听

小程序显示)、onHide(监听小程序隐藏)等,在不同阶段、不同场景可以使用不同的生命周期函数。此外,App.js中还可以定义一些全局的函数和数据,其他页面引用App.js文件后就可以直接使用全局函数和数据,如图2.2所示。

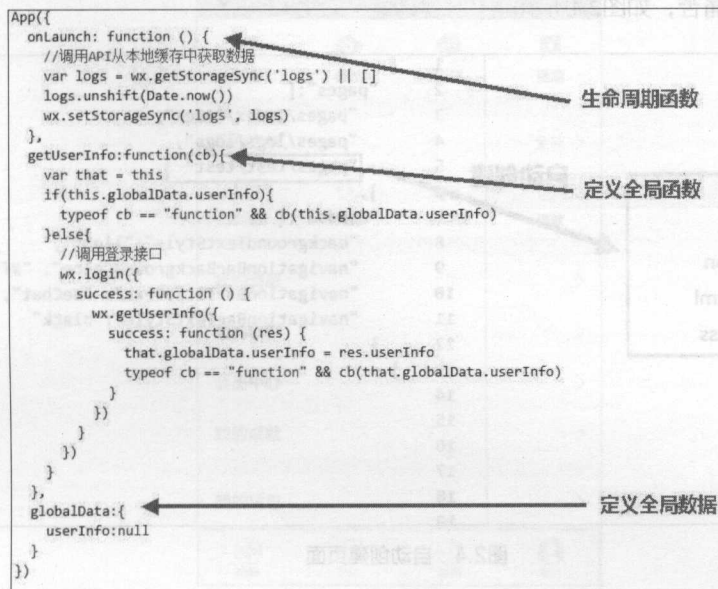


图2.2 App.js小程序逻辑

2. App.json小程序公共设置

App.json文件可以对5个功能进行设置:配置页面路径、配置窗口表现、配置标签导航、配置网络超时、配置debug模式,如图2.3所示。

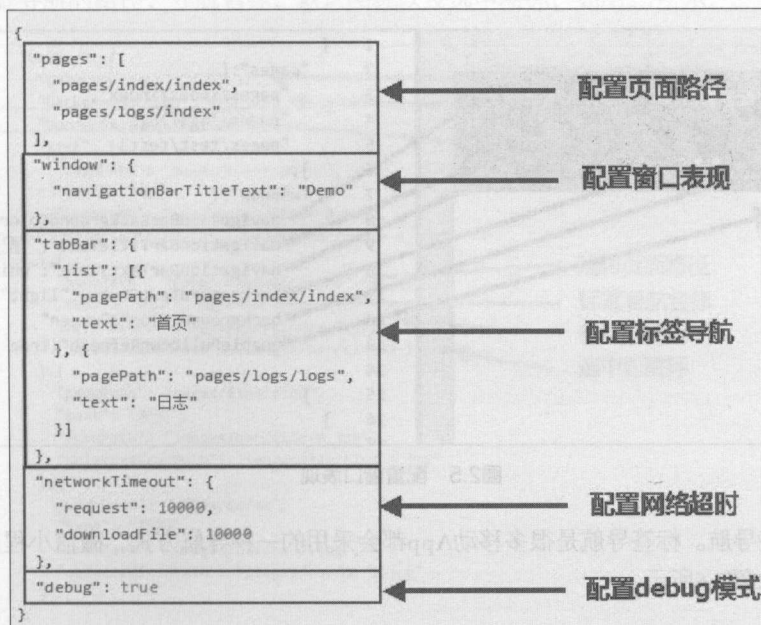


图2.3 App.json的5个功能

1 配置页面路径。页面路径定义了一个数组，存放多个页面的访问路径，它是进行页面访问的必要条件。如果在这里没有配置页面的访问路径，页面被访问时就会报错；如果在这里定义了页面的访问路径，微信小程序框架就可以在页面文件夹下建立相应名称的文件夹以及文件，免去手动添加文件夹和文件的痛苦，如图2.4所示。

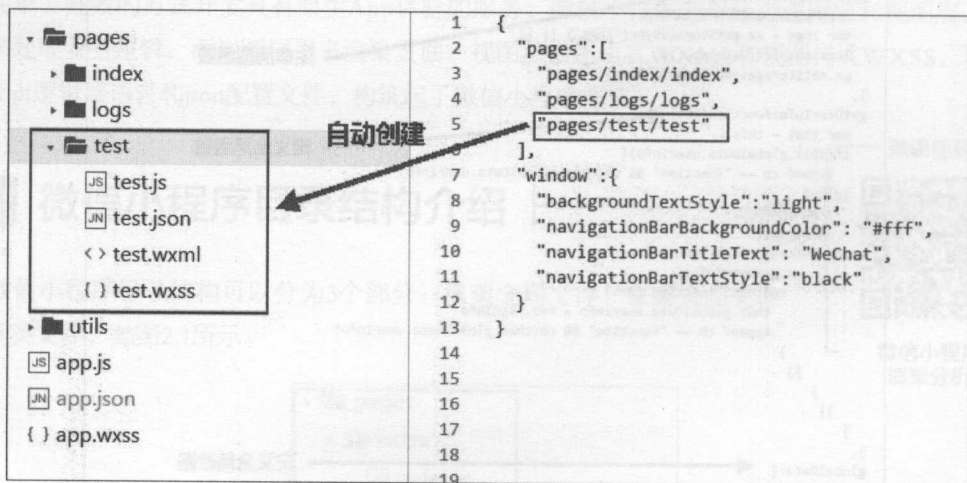


图2.4 自动创建页面

2 配置窗口表现。窗口表现用于配置小程序的状态栏、导航条、标题、窗口背景色，可以设置导航条背景色（navigationBarBackgroundColor）、导航条文字（navigationBarTitleText）和导航条文字颜色（navigationBarTextStyle）；还可以设置窗口是否可以下拉刷新（enablePullDownRefresh）、默认值是否可以下拉刷新、窗口的背景色（backgroundColor）和下拉背景字体或者loading样式（backgroundTextStyle），如图2.5所示。

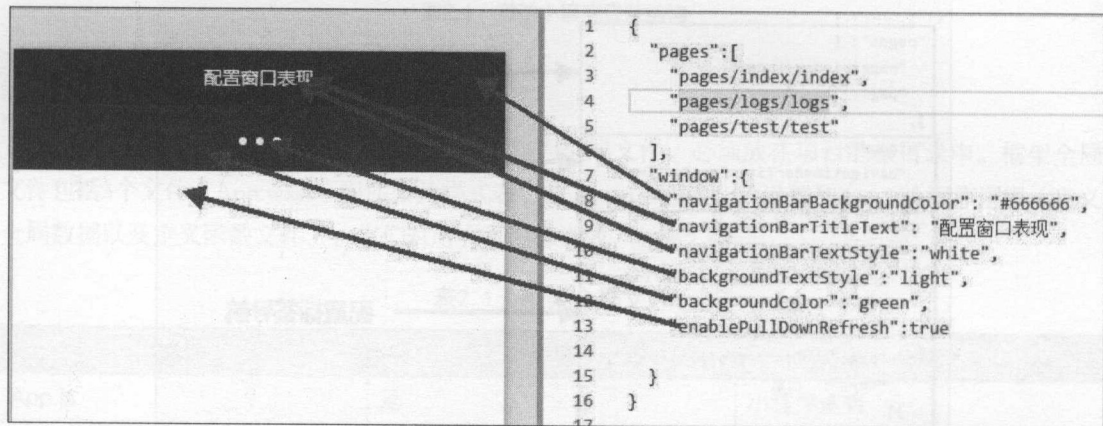


图2.5 配置窗口表现

3 配置标签导航。标签导航是很多移动App都会采用的一种导航方式，微信小程序同样可以实现这样的效果，如图2.6所示。



图2.6 猫眼电影App标签导航

怎么制作标签导航呢？我们需要在App.json里配置tabBar属性。tabBar是一个对象，可以配置标签导航文字的默认颜色、选中颜色，标签导航背景色以及上边框颜色，上边框颜色现在可以配置两种颜色black/white。标签导航存放放到list数组里面，list里的每个对象，对应一个标签导航，每个对象里可以配置标签导航的路径、导航名称、默认图标以及选中图标，如图2.7所示。

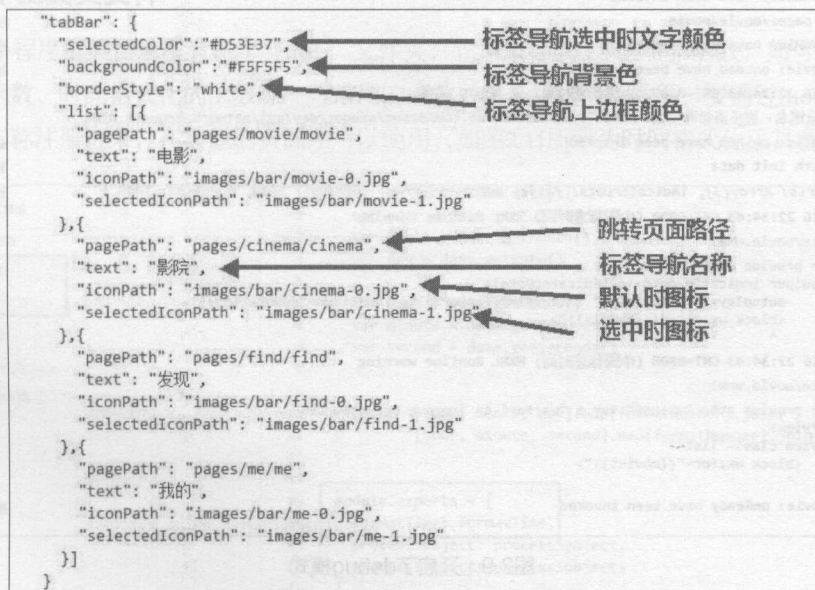


图2.7 猫眼电影微信小程序标签导航配置

4 配置网络超时。可以配置网络请求、文件上传、文件下载时最大的请求时间，超过这个时间，则不再请求。

5 配置debug模式。配置debug模式可方便微信小程序开发者调试开发程序，如图2.8和图2.9所示为没有开启debug模式和开启了debug模式的调试信息对比。

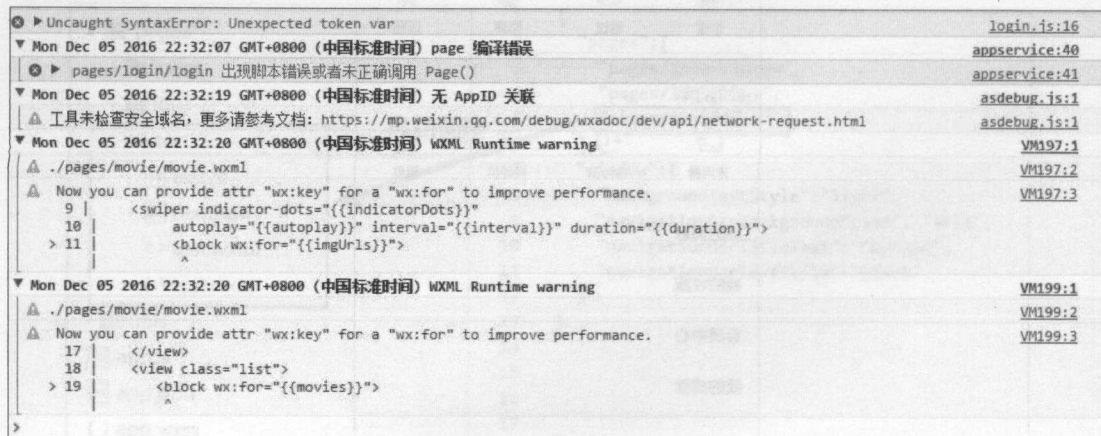


图2.8 没有开启debug模式

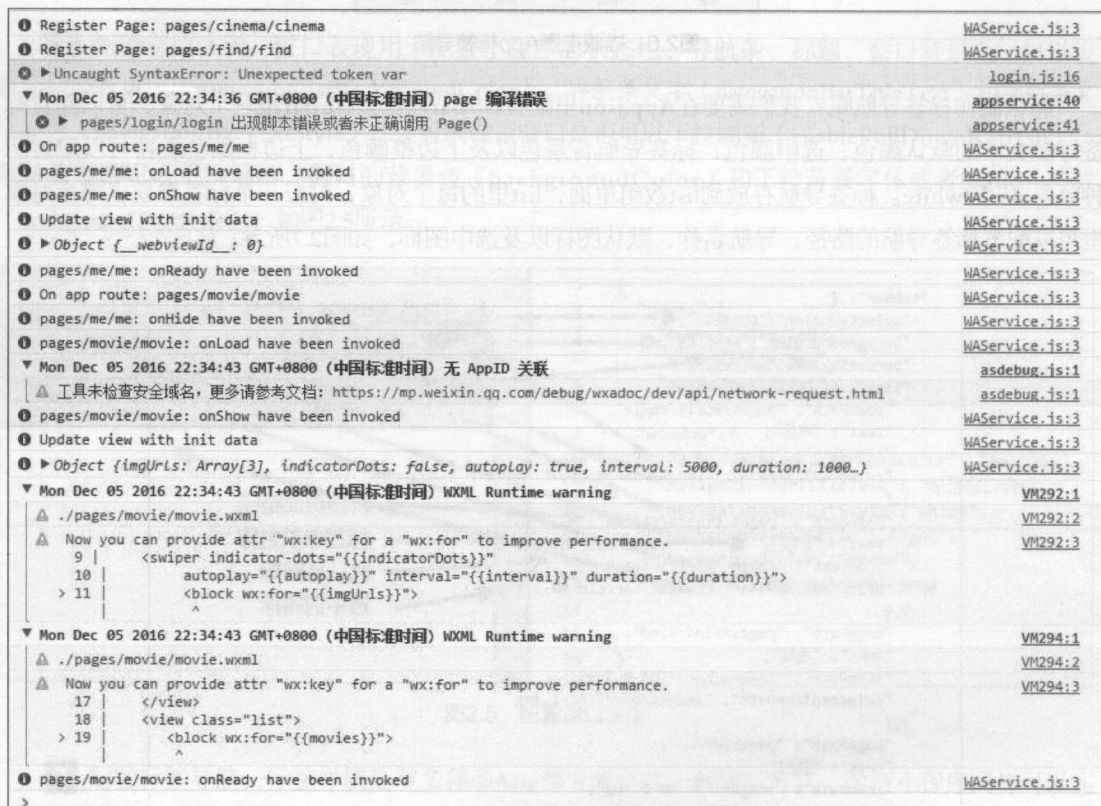


图2.9 开启了debug模式

从图2.8和图2.9可以看出,开启debug模式后,可以看到每一步的调用情况、访问路径以及错误信息,这样更加方便开发者进行调试工作。

App.json作为全局配置文件提供决定页面文件路径、配置窗口的表现、配置底部标签导航、配置网络连接超时、配置debug模式开发的功能,配置也比较容易。

3. App.wxss小程序公共样式表

App.wxss文件对CSS样式进行了扩展,和CSS的使用方式一样,类选择器和行内样式的写法兼容大部分CSS样式,有一些CSS样式在这里是不起作用的,同时它还扩展了CSS,形成了自己风格的样式文件,是对所有页面定义的一个全局样式。只要页面有全局样式里的class,就都可以渲染全局样式里的效果;但如果页面又重新定义了这个class样式,则会把全局样式覆盖掉,而使用自己的样式。小程序公共样式表如图2.10所示。

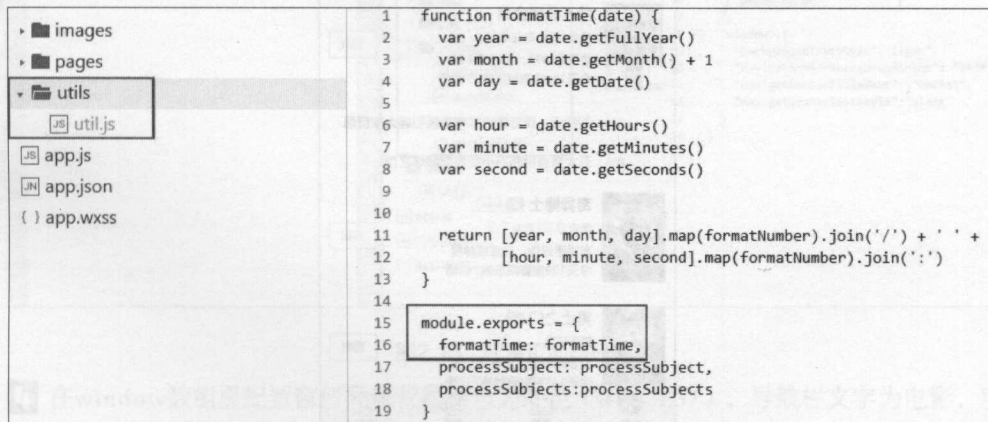
```
/**app.wxss**/  
.container {  
  height: 100%;  
  display: flex;  
  flex-direction: column;  
  align-items: center;  
  justify-content: space-between;  
  padding: 200rpx 0;  
  box-sizing: border-box;  
}
```

图2.10 小程序公共样式表

除了App.wxss提供的默认全局样式,用户自己也可以定义一些全局样式,这样方便每个页面的使用,又不用在每个页面上都写一次,达到一次定义、其他页面直接引用的复用效果。

2.1.2 工具类文件

在微信小程序框架目录里有一个“utils”文件夹,它用来存放工具栏的js函数,如可以放置一些日期格式化的函数、时间格式化的函数等一些常用的函数。定义完这些函数后,要通过module.exports将定义的函数名称注册进来,在其他的页面才可以使用,如图2.11所示为时间格式化工具类文件。



```
1 function formatTime(date) {  
2   var year = date.getFullYear()  
3   var month = date.getMonth() + 1  
4   var day = date.getDate()  
5  
6   var hour = date.getHours()  
7   var minute = date.getMinutes()  
8   var second = date.getSeconds()  
9  
10  
11   return [year, month, day].map(formatNumber).join('/') + ' ' +  
12     [hour, minute, second].map(formatNumber).join(':')  
13 }  
14  
15 module.exports = {  
16   formatTime: formatTime,  
17   processSubject: processSubject,  
18   processSubjects: processSubjects  
19 }
```

图2.11 utils.js工具类文件

2.1.3 框架页面文件

一个小程序框架页面文件由4个文件组成，分别是js页面逻辑、wxml页面结构、wxss页面样式表和json页面配置，如表2.2所示。

表2.2 框架页面文件

| 文件类型 | 必填 | 作用 |
|------|----|-------|
| js | 是 | 页面逻辑 |
| wxml | 是 | 页面结构 |
| wxss | 否 | 页面样式表 |
| json | 否 | 页面配置 |

微信小程序的框架页面文件都是放置在“pages”文件夹下面的，如图2.12所示。

每个页面都有一个独立的文件夹，就像日志页面的“logs”文件夹，它的下面放置4个文件：logs.js可进行业务路径处理；logs.json是页面的配置，可以覆盖全局App.json配置；logs.wxml是页面结构，负责渲染页面；logs.wxss是针对logs.wxml页面的样式文件。

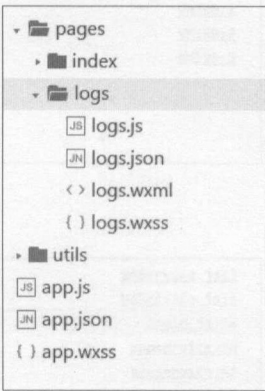


图2.12 页面文件

2.1.4 小试牛刀：制作猫眼电影底部标签导航

猫眼电影底部标签导航分为4个标签导航：电影、影院、发现、我的，如图2.13所示。



图2.13 猫眼电影底部标签导航

1 新建一个无AppID的movie项目，如图2.14所示。

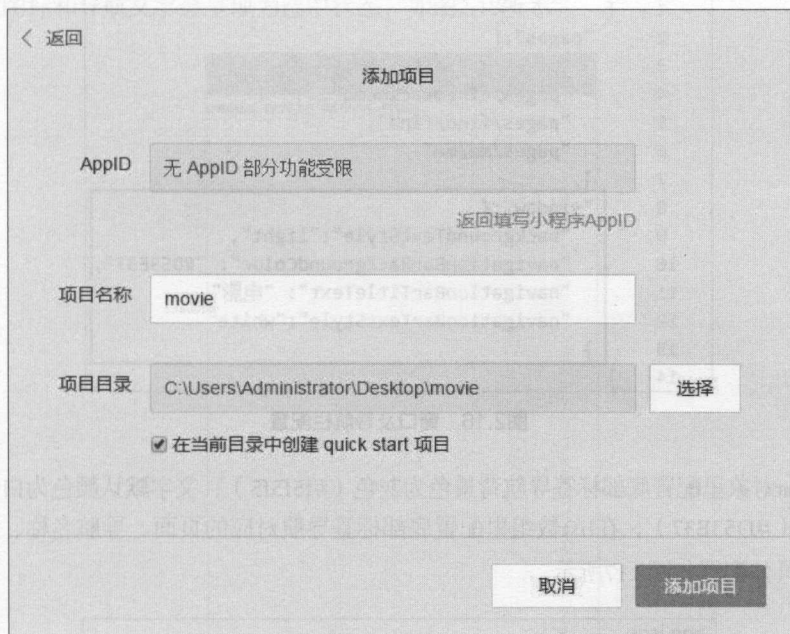


图2.14 添加项目

2 将准备好的底部标签导航图标拷贝到movie项目下面。

3 打开App.json配置文件，在pages数组里添加4个页面路径：电影“pages/movie/movie”、影院“pages/cinema/cinema”、发现“pages/find/find”、我的“pages/me/me”，保存后会自动生成相应的页面文件夹；删除“pages/index/index”“pages/logs/logs”页面路径以及对应的文件夹，如图2.15所示。

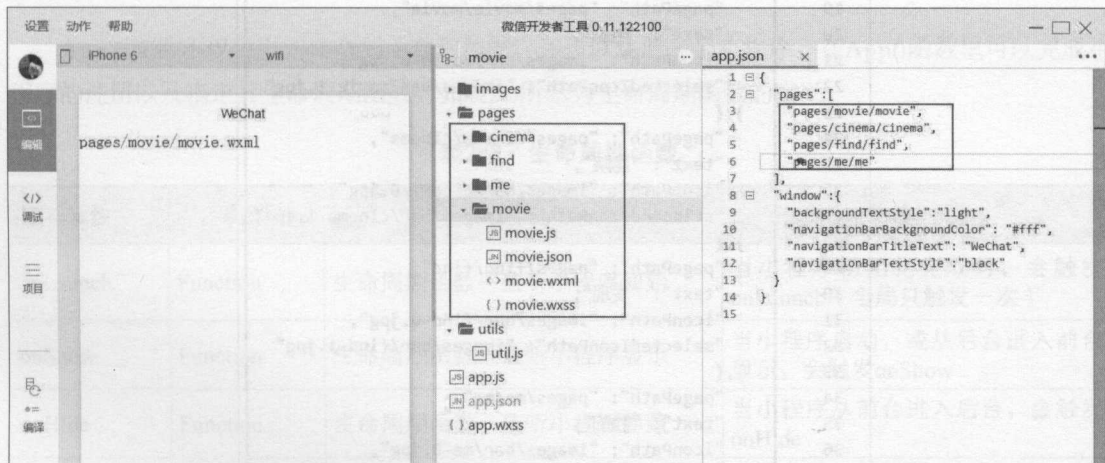


图2.15 配置页面路径

4 在window数组里配置窗口导航背景颜色为红色（#D53E37），导航栏文字为电影，字体颜色为白色（white），具体配置如图2.16所示。

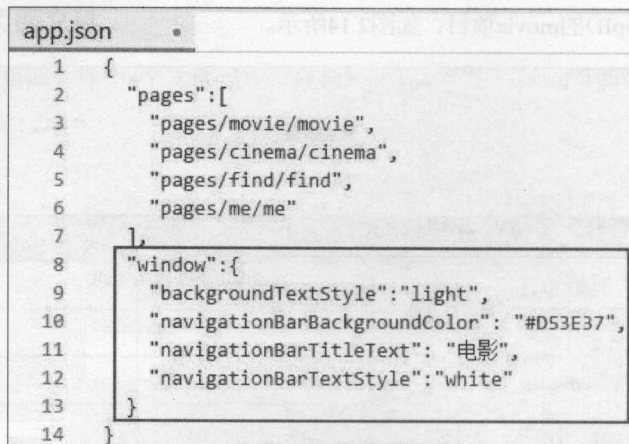


图2.16 窗口及导航栏配置

5 在tabBar对象里配置底部标签导航背景色为灰色（#f5f5f5），文字默认颜色为白色（white），选中时为红色（#D53E37），在list数组里配置底部标签导航对应的页面、导航名称、默认时图标、选中时图标，具体配置如图2.17所示。

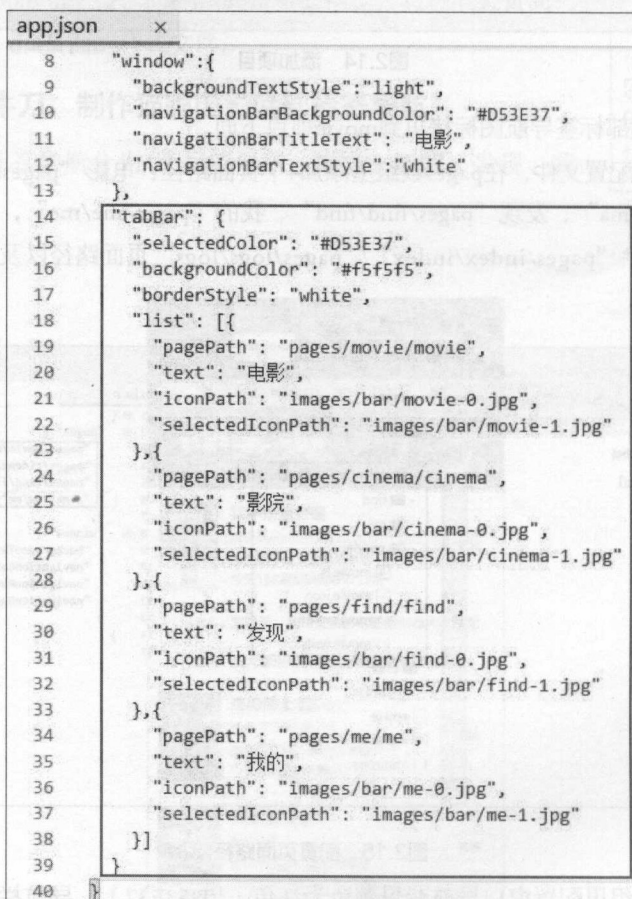


图2.17 底部标签导航配置

这样就完成了猫眼电影底部标签导航的配置，单击不同的导航标签，可以切换显示不同的页面，同时导航图标和导航文字会呈现为选中状态，如图2.18所示。

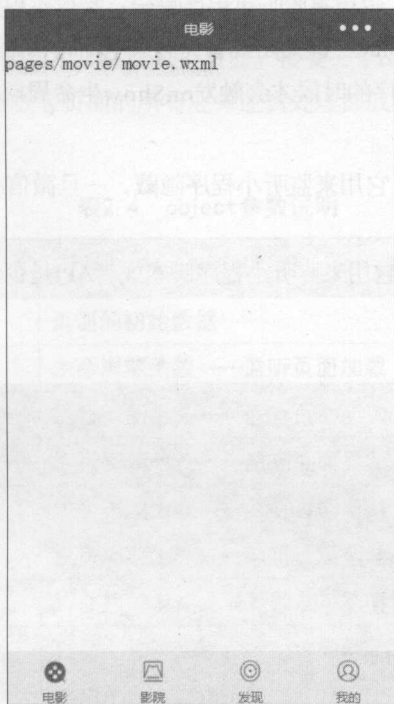


图2.18 电影界面

2.2 微信小程序注册程序的应用

App.js文件不仅可以定义全局函数和数据，还可以注册一个小程序。在App0函数里可以完成小程序的注册以及指定其生命周期函数。如表2.3所示为生命周期函数的定义。

表2.3 生命周期函数

| 属性 | 类型 | 描述 | 触发时机 |
|----------|----------|----------------|---|
| onLaunch | Function | 生命周期函数—监听小程序初始 | 当小程序初始化完成时，会触发onLaunch（全局只触发一次） |
| onShow | Function | 生命周期函数—监听小程序显示 | 当小程序启动，或从后台进入前台显示，会触发onShow |
| onHide | Function | 生命周期函数—监听小程序隐藏 | 当小程序从前台进入后台，会触发onHide |
| onError | Function | 错误监听函数 | 当小程序发生脚本错误，或者api调用失败时，会触发onError并带上错误信息 |
| 其他 | Any | | 开发者可以添加任意函数或数据到Object参数中，用this可以使用 |

1 onLaunch生命周期函数。它用来监听小程序初始化，一旦初始化完成，就会触发该函数，这个生命周期函数只会触发一次。

2 onShow生命周期函数。它用来监听小程序显示。微信小程序有前后台定义。当用户单击左上角的“关闭”按钮或者按“Home”键关闭或者突然来电话时，微信小程序都没有销毁，而是进入后台；当再次进入微信或者小程序的时候才会触发onShow生命周期函数。只要程序启动或者从后台进入到前台都会触发该函数。

3 onHide生命周期函数。它用来监听小程序隐藏，一旦微信小程序从前台进入到后台，就会触发该函数。

4 onError生命周期函数。它用来监听小程序脚本或者API是否发生错误，发生错误时返回错误信息。

示例代码如下。

```
App({
  onLaunch: function() {
    // Do something initial when launch.
  },
  onShow: function() {
    // Do something when show.
  },
  onHide: function() {
    // Do something when hide.
  },
  onError: function(msg) {
    console.log(msg)
  },
  globalData: 'I am global data'
})
```

在页面里调用App.js全局数据：

在页面的js文件里，只需要写下面两句话就可以调用到全局数据globalData。

```
var AppInstance = getApp()
console.log(AppInstance.globalData)
```

不仅可以调用全局数据，还可以调用自定义的全局函数，但是不要调用声明周期函数。



注意：

- (1) App() 必须在 App.js 中注册，且不能注册多个。
- (2) 不要在定义于 App() 内的函数中调用 getApp()，使用 this 就可以拿到 App 实例。
- (3) 不要在 onLaunch 的时候调用 getCurrentPage()，此时 page 还没有生成。
- (4) 通过 getApp() 获取实例之后，不要私自调用生命周期函数。

2.3 微信小程序注册页面的使用

在每个页面文件夹里，都有一个页面对应的js文件，就像日志“logs”文件夹，对应的就是logs.js文件。在这个文件里的Page() 函数用于注册一个页面。接受一个object 参数，其指定页面的初始数据、生命周期函数、事件处理函数等页面的所有业务逻辑处理都放在这个文件里。object参数说明如表2.4所示。

表2.4 object参数说明

| 属性 | 类型 | 描述 |
|-------------------|----------|---|
| data | Object | 页面的初始数据 |
| onLoad | Function | 生命周期函数——监听页面加载 |
| onReady | Function | 生命周期函数——监听页面初次渲染完成 |
| onShow | Function | 生命周期函数——监听页面显示 |
| onHide | Function | 生命周期函数——监听页面隐藏 |
| onUnload | Function | 生命周期函数——监听页面卸载 |
| onPullDownRefresh | Function | 页面相关事件处理函数——监听用户下拉动作 |
| onReachBottom | Function | 页面上拉触底事件的处理函数 |
| onShareAppMessage | Function | 用户点击右上角分享 |
| 其他 | Any | 开发者可以添加任意函数或数据到 object 参数中，在页面的函数中用 this 可以访问 |

Page()函数的使用代码如下。

```
Page({
  data: {
    text: "This is page data."
  },
  onLoad: function(options) {
    // Do some initialize when page load.
  },
  onReady: function() {
    // Do something when page ready.
  },
  onShow: function() {
    // Do something when page show.
  },
  onHide: function() {
    // Do something when page hide.
  },
  onUnload: function() {
    // Do something when page close.
  },
  onPullDownRefresh: function() {
```



```

    // Do something when pull down.
  },
  onReachBottom: function() {
    // Do something when page reach bottom.
  },
  onShareAppMessage: function () {
    // return custom share data when user share.
  },
  // Event handler.
  viewTap: function() {
    this.setData({
      text: 'Set some data for updating view.'
    })
  },
},
customData: {
  hi: 'MINA'
}
})

```

2.3.1 页面初始化数据

data 页面初始化数据：初始化数据将作为页面的第一次渲染。data 将会以 JSON 的形式由逻辑层传至渲染层，所以其数据必须可以转换成 JSON 的格式——字符串、数字、布尔值、对象或数组。渲染界面可以通过 WXML 对数据进行绑定。

示例代码如下。

```
<text class="user-motto">{{motto}}</text>
```

```

Page({
  data: {
    motto: 'Hello World',
    userInfo: {}
  })
}

```

2.3.2 生命周期函数

1 onLoad 页面加载。一个页面只会调用一次，接收页面参数可以获取 wx.navigateTo 和 wx.redirectTo 及 <navigator/> 中的 query。

2 onShow 页面显示。每次打开页面都会调用一次。

3 onReady 页面初次渲染完成。一个页面只会调用一次，代表页面已经准备妥当，可以和视图层进行交互，对界面的设置如 wx.setNavigationBarTitle 请在 onReady 之后设置。

4 onHide 页面隐藏。当 navigateTo 或底部 Tab 切换时调用。

5 onUnload 页面卸载。当 redirectTo 或 navigateBack 时调用。

2.3.3 页面相关事件处理函数

1 onPullDownRefresh 下拉刷新。监听用户下拉刷新事件，需要在 config 的 window 选项中开启 enablePullDownRefresh。当处理完数据刷新后，wx.stopPullDownRefresh 可以停止当前页面的下拉

刷新。

2 onShareAppMessage用户分享。只有定义了此事件处理函数，右上角菜单才会显示“分享”按钮，用户单击“分享”按钮的时候会调用，此事件需要 return 一个 Object，用于自定义分享内容。分享参数说明如表2.5所示。

表2.5 分享参数说明

| 字段 | 说明 | 默认值 |
|-------|------|--------------------------|
| title | 分享标题 | 当前小程序名称 |
| desc | 分享描述 | 当前小程序名称 |
| path | 分享路径 | 当前页面 path，必须是以 / 开头的完整路径 |

示例代码如下。

```
Page({
  onShareAppMessage: function () {
    return {
      title: '自定义分享标题',
      desc: '自定义分享描述',
      path: '/page/user?id=123'
    }
  }
})
```

2.3.4 页面路由管理

微信小程序的页面路由都是有微信小程序框架来管理的，路由的触发方式及页面生命周期函数如表2.6所示。

表2.6 路由的触发方式及页面生命周期函数

| 页面路由方式 | 触发时机 | 路由后页面 | 路由前页面 |
|--------|--|--|---|
| 初始化 | 小程序打开的第一个页面 | onLoad, onShow | |
| 打开新页面 | 调用 API wx.navigateTo 或使用组件 <navigator open-type="navigate"/> | onLoad, onShow | onHide |
| 页面重定向 | 调用 API wx.redirectTo 或使用组件 <navigator open-type="redirect"/> | onLoad, onShow | onUnload |
| 页面返回 | 调用 API wx.navigateBack 或用户按左上角的“返回”按钮 | onShow | onUnload (多层页面返回每个页面都会按顺序触发 onUnload) |
| Tab 切换 | Function | 调用 API wx.swit-chTab 或使用组件<navigator open-type="switchTab"/> 或用户切换 Tab | |

2.3.5 自定义函数

除了初始化数据和生命周期函数，Page 中还可以定义一些特殊的函数：事件处理函数。在渲染层可以在组件中加入事件绑定，当达到触发事件时，就会执行 Page 中定义的事件处理函数。

示例代码如下。

```
<view bindtap="clickMe"> click me </view>
```

```
Page({
  clickMe: function() {
    console.log('view tap')
  }
})
```

2.3.6 setData设置值函数

Page.prototype.setData() 设置值函数：setData 函数用于将数据从逻辑层发送到视图层，同时改变对应的 this.data 的值。

setData() 参数格式：接受一个对象，以 key、value 的形式表示将 this.data 中的 key 对应的值改变成 value。

其中，key 可以非常灵活，以数据路径的形式给出，如 array[2].message, a.b.c.d，并且不需要在 this.data 中预先定义。

示例代码如下。

```
<!--index.wxml-->
<view>{{text}}</view>
<button bindtap="changeText"> Change normal data </button>
<view>{{array[0].text}}</view>
<button bindtap="changeItemInArray"> Change Array data </button>
<view>{{object.text}}</view>
<button bindtap="changeItemInObject"> Change Object data </button>
<view>{{newField.text}}</view>
<button bindtap="addNewField"> Add new data </button>
```

```
//index.js
Page({
  data: {
    text: 'init data',
    array: [{text: 'init data'}],
    object: {
      text: 'init data'
    }
  },
  changeText: function() {
    // this.data.text = 'changed data' // bad, it can not work
    this.setData({
      text: 'changed data'
    })
  }
})
```



```

},
changeItemInArray: function() {
  // you can use this way to modify a danamic data path
  this.setData({
    'array[0].text': 'changed data'
  })
},
changeItemInObject: function(){
  this.setData({
    'object.text': 'changed data'
  });
},
addNewField: function() {
  this.setData({
    'newField.text': 'new data'
  })
}
})

```



注意：直接修改 `this.data` 无效，无法改变页面的状态，还会造成数据不一致。单次设置的数据不能超过1 024kB，请尽量避免一次设置过多的数据。

2.4 微信小程序如何绑定数据

WXML页面里的动态数据都是来自js文件Page的data，数据绑定就是通过双大括号（`{{}}`）将变量包起来，在WXML页面里将数据值显示出来。

示例代码如下。

```

index.wxml
<view> {{ message }} </view>

```

```

index.js
Page({
  data: {
    message: 'Hello MINA!'
  }
})

```

2.4.1 组件属性绑定

组件属性绑定是将data里的数据绑定到微信小程序的组件上，示例代码如下。

```

<view id="item-{{id}}"> </view>

```

```

Page({
  data: {
    id: 0
  }
})

```

```
}  
})
```

2.4.2 控制属性绑定

控制属性绑定用来进行if语句条件判断，如果满足条件，则执行，否则不执行，示例代码如下。

```
<view wx:if="{{condition}}"> </view>
```

```
Page({  
  data: {  
    condition: true  
  }  
})
```

2.4.3 关键字绑定

关键字绑定常用于组件的一些关键字，像复选框组件一样。checked关键字如果等于true则代表选中复选框，false则代表不选中复选框，示例代码如下。

```
<checkbox checked="{{false}}"> </checkbox>
```

不要直接写 checked="false"，其计算结果是一个字符串，转换成 boolean 类型后代表真值。

2.4.4 运算

可以在 {{}} 内进行简单的运算，支持以下几种方式的运算。

1. 三元运算

```
<view hidden="{{flag ? true : false}}"> Hidden </view>
```

2. 数学运算

```
<view> {{a + b}} + {{c}} + d </view>
```

```
Page({  
  data: {  
    a: 1,  
    b: 2,  
    c: 3  
  }  
})
```

view中的内容为 3 + 3 + d。

3. 逻辑判断

```
<view wx:if="{{length > 5}}"> </view>
```

4. 字符串运算

```
<view>{{"hello" + name}}</view>
```

```
Page({  
  data:{
```

```
name: 'MINA'
}
})
```

5. 数据路径运算

```
<view>{{object.key}} {{array[0]}}</view>
```

```
Page({
  data: {
    object: {
      key: 'Hello '
    },
    array: ['MINA']
  }
})
```

2.4.5 小试牛刀：天气微信小程序

天气微信小程序用来显示天气的温度、最低温度、最高温度及天气情况，下面通过数据绑定的方式来显示天气情况，如图2.19所示。

微课视频



天气微信小程序



图2.19 天气微信小程序

1 创建一个无AppID的weather项目，如图2.20所示。

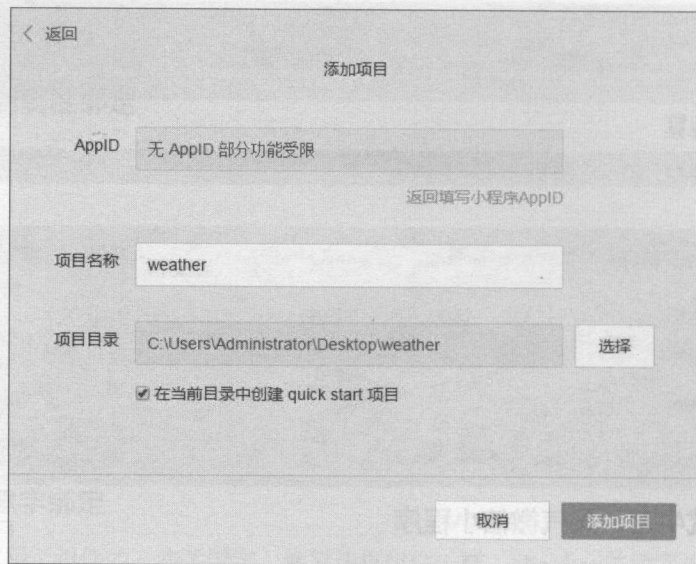


图2.20 weather项目

2 进入index.wxml、index.js、index.wxss文件，清空所有内容，进入App.json，修改导航栏标题为“中国天气网”。

3 进入index.wxml，进行当天天气情况的界面布局，包括温度、最低温、最高温、天气情况、城市、星期、风行情况，代码如下。

```
<view class="content">
  <view class="today">
    <view class="info">
      <view class="temp">℃ </view>
      <view class="lowhigh"></view>
      <view class="type"></view>
      <view class="city"></view>
      <view class="week"></view>
      <view class="weather"></view>
    </view>
  </view>
</view>
```

4 进入index.js，在data里提供天气的数据，让这些数据在界面里显示出来，代码如下。

```
Page({
  data:{
    temp:"4",
    low:"-1℃ ",
    high:"10℃ ",
    type:" 晴 ",
    city:" 北京 ",
    week:" 星期二 ",
    weather:" 无持续风行 微风级 "
  }
})
```

5 进入index.wxml，将data里提供的天气数据绑定到页面里，代码如下。

```
<view class="content">
  <view class="today">
    <view class="info">
      <view class="temp">{{temp}}℃ </view>
      <view class="lowhigh">{{low}}/{{high}}</view>
      <view class="type">{{type}}</view>
      <view class="city">{{city}}</view>
      <view class="week">{{week}}</view>
      <view class="weather"> {{weather}} </view>
    </view>
  </view>
</view>
```

界面效果如图2.21所示。

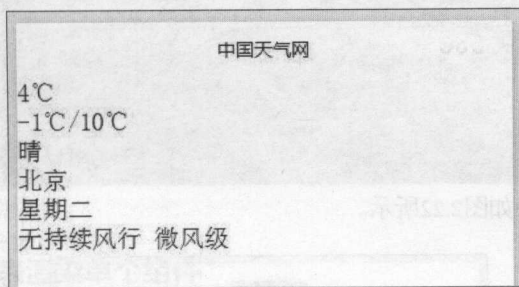


图2.21 天气界面效果

6 进入index.wxss，将index.wxml添加样式，美化页面，代码如下。

```
.content{
  font-family : 微软雅黑 , 宋体 ;
  font-size: 14px;
  background-size:cover;
  height: 100%;
  width:100%;
  color:#333333;
}
.today{
  padding-top:70rpx;
  height:50%;
}
.temp{
  font-size:80px;
  text-align: center;
}
.city{
  font-size: 20px;
  text-align: center;
  margin-top:20rpx;
  margin-right: 10rpx;
}
```

```
.lowhigh{
  font-size: 12px;
  text-align: center;
  margin-top: 30rpx;
}
.type{
  font-size: 16px;
  text-align: center;
  margin-top: 30rpx;
}
.week{
  font-size: 12px;
  text-align: center;
  margin-top: 30rpx;
}

.weather{
  font-size: 12px;
  text-align: center;
  margin-top: 20rpx;
}
```

添加样式后的界面效果如图2.22所示。

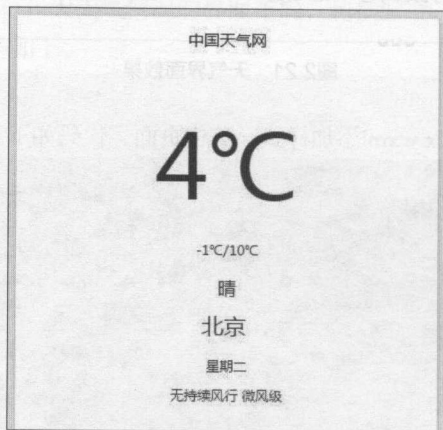


图2.22 添加样式后的界面效果

将js里的data通过数据绑定的方式就可以在wxml里通过双大括号的方式将数据值显示出来，动态地加载数据，以实现数据绑定的动态效果。

2.5 微信小程序条件渲染

2.5.1 wx:if 判断单个组件

在微信小程序框架里，使用 `wx:if="{{condition}}"` 来判断是否需要渲染该代码块，示例代码如下。


```
<view wx:if="{{condition}}"> True </view>
```

使用 `wx:elif` 和 `wx:else` 来添加一个 `else` 块，示例代码如下。

```
<view wx:if="{{length > 5}}"> 1 </view>
<view wx:elif="{{length > 2}}"> 2 </view>
<view wx:else> 3 </view>
```

2.5.2 block wx:if 判断多个组件

因为 `wx:if` 是一个控制属性，需要将它添加到一个标签上。但是，如果我们想一次性判断多个组件标签，则可以使用一个 `<block/>` 标签将多个组件包装起来，并在其上使用 `wx:if` 控制属性，示例代码如下。

```
<block wx:if="{{true}}">
  <view> view1 </view>
  <view> view2 </view>
</block>
```

`<block/>` 并不是一个组件，仅仅是一个包装元素，不会在页面中做任何渲染，只接受控制属性。

2.6 微信小程序列表渲染

2.6.1 wx:for 列表渲染单个组件

在组件上使用 `wx:for` 控制属性绑定一个数组，即可使用数组中的各项数据重复渲染该组件。默认数组当前项的下标变量名默认为 `index`，数组当前项的变量名默认为 `item`，示例代码如下。

```
<view wx:for="{{array}}">
  {{index}}: {{item.message}}
</view>
```

```
Page({
  data: {
    array: [{
      message: 'foo',
    }, {
      message: 'bar'
    }]
  }
})
```

使用 `wx:for-item` 可以指定数组当前元素的变量名，使用 `wx:for-index` 可以指定数组当前下标的变量名，示例代码如下。

```
<view wx:for="{{array}}" wx:for-index="idx" wx:for-item="itemName">
  {{idx}}: {{itemName.message}}
</view>
```

2.6.2 block wx:for 列表渲染多个组件

`wx:for` 应用在某一个组件上，但是如果渲染一个包含多节点的结构块，`wx:for` 就需要应用在

<block/>标签上，示例代码如下。

```
<block wx:for="{{[1, 2, 3]}}">
  <view> {{index}}: </view>
  <view> {{item}} </view>
</block>
```

2.6.3 wx:key 指定唯一标识符

如果列表中项目的位置会动态改变或者有新的项目添加到列表中，并且希望列表中的项目保持自己的特征和状态（如 <input/> 中的输入内容、<switch/> 的选中状态），需要使用 wx:key 来指定列表中项目的唯一标识符。

wx:key 的值有以下两种形式。

1 字符串。字符串代表在 for 循环的 array 中 item 的某个 property，该 property 的值需要是列表中唯一的字符串或数字，且不能动态改变。

2 保留关键字。*this 代表在 for 循环中的 item 本身，这种表示需要 item 本身是一个唯一的字符串或者数字，当数据改变触发渲染层重新渲染的时候，会校正带有 key 的组件，框架会确保它们被重新排序，而不是重新创建，以确保使组件保持自身的状态，并且提高列表渲染时的效率。

示例代码如下。

```
<switch wx:for="{{objectArray}}" wx:key="unique" style="display: block;"> {{item.id}} </switch>
Page({
  data: {
    objectArray: [
      {id: 5, unique: 'unique_5'},
      {id: 4, unique: 'unique_4'},
      {id: 3, unique: 'unique_3'},
      {id: 2, unique: 'unique_2'},
      {id: 1, unique: 'unique_1'},
      {id: 0, unique: 'unique_0'},
    ]
  }
})
```



注意：如不提供 wx:key，会报一个 warning，如果明确地知道该列表是静态，或者不必关注其顺序，可以选择忽略。

2.7 微信小程序定义模板

WXML提供模板（template）功能，可以在模板中对一些共用的、复用的代码定义代码片段，然后在不同的地方调用，以达到一次编写、多次直接使用的效果。

2.7.1 定义模板

在<template/>内定义代码片段，使用name属性作为模板的名字，示例代码如下。

```
<template name="msgItem">
  <view>
    <text> {{index}}: {{msg}} </text>
    <text> Time: {{time}} </text>
  </view>
</template>
```

2.7.2 使用模板

在WXML文件里，使用 `is` 属性，声明需要使用的模板，然后将模板所需要的 `data` 传入，示例代码如下。

```
<template is="msgItem" data="{{item}}" />
Page({
  data: {
    item: {
      index: 0,
      msg: 'this is a template',
      time: '2016-09-15'
    }
  }
})
```

`is` 属性可以使用三元运算语法来动态地决定具体需要渲染哪个模板，示例代码如下。

```
<template name="odd">
  <view> odd </view>
</template>
<template name="even">
  <view> even </view>
</template>

<block wx:for="{{[1, 2, 3, 4, 5]}}">
  <template is="{{item % 2 == 0 ? 'even' : 'odd'}}" />
</block>
```

2.8 微信小程序的引用功能

WXML 提供两种文件引用方式：`import`和`include`。两者的区别在于，`import`引用模板文件，`include`将整个文件除了`<template/>`之外进行引用。

2.8.1 import引用

`import`可以在该文件中使用目标文件定义的`template`。

假如在 `item.wxml` 中定义了一个叫`item`的`template`，则示例代码如下。

```
<!-- item.wxml -->
<template name="item">
  <text>{{text}}</text>
</template>
```

在 `index.wxml` 中引用了 `item.wxml`，就可以使用`item`模板，示例代码如下。


```
<import src="item.wxml"/>
<template is="item" data="{{text: 'forbar'}}"/>
```

2.8.2 include引用

include可以将目标文件除了<template/>之外的整个代码引入，相当于是复制到include位置，示例代码如下。

```
<!-- index.wxml -->
<include src="header.wxml"/>
<view> body </view>
<include src="footer.wxml"/>
```

```
<!-- header.wxml -->
<view> header </view>
```

```
<!-- footer.wxml -->
<view> footer </view>
```

2.9 沙场大练兵：仿香哈菜谱微信小程序

香哈菜谱是围绕美食而做的一款小程序，在这里可以查看各式各样的菜谱以及制作方法。对于菜谱类App软件，用户使用的频次不高。当用户碰到不会做的菜式或者想做一些新的菜式时，才会去App软件查看，而微信小程序就可以满足这种低频使用的需要，如图2.23和图2.24所示。



图2.23 学做菜



图2.24 头条

2.9.1 底部标签导航设计

仿香哈菜谱微信小程序的底部有5个导航标签：学做菜、头条、美食圈、消息、我的，标签导航选中时导航图标和导航文字都会变为红色，如图2.25所示。

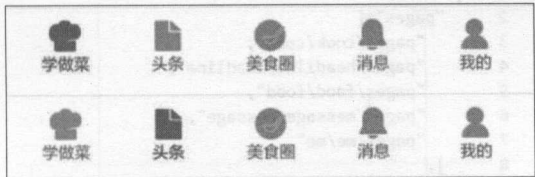


图2.25 底部标签导航选中效果

1 新建一个无AppID的xhcp项目，如图2.26所示。

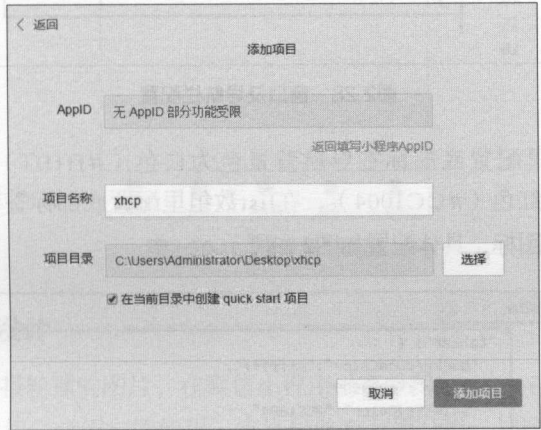


图2.26 添加项目

2 将准备好的底部标签导航图标、美食轮播图片、宫格导航图标、香哈头条美食图片复制到“pages”文件夹下。

3 打开App.json配置文件，在pages数组里添加5个页面路径：“pages/cook/cook” “pages/headline/headline” “pages/food/food” “pages/message/message” “pages/me/me”，保存后会自动生成相应的页面文件夹；删除“pages/index/index” “pages/logs/logs” 页面路径以及对应的文件夹，如图2.27所示。

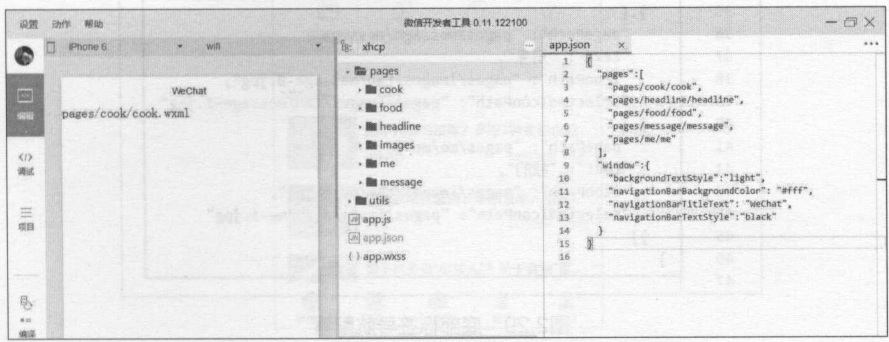


图2.27 配置页面路径

4 在window数组里配置窗口导航背景颜色为灰色（#494949），导航栏文字为“学做菜”，字体颜色为白色（#ffffff），具体配置如图2.28所示。

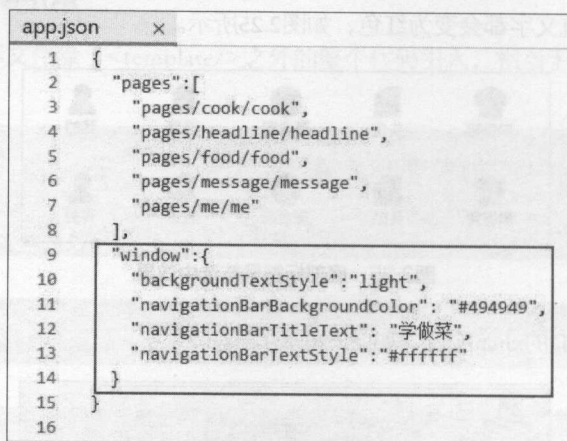


图2.28 窗口及导航栏配置

5 在tabBar对象里配置底部标签导航背景颜色为白色（#ffffff），文字默认颜色为灰色（#999999），选中时为红色（#CC1004），在list数组里配置底部标签导航对应的页面、导航名称、默认时图标、选中时图标，具体配置如图2.29所示。

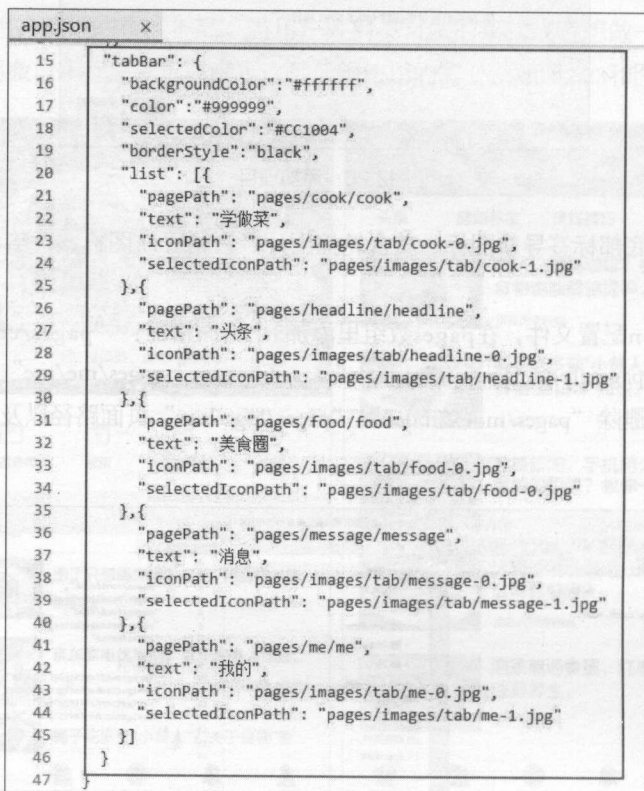


图2.29 底部标签导航配置

这样就完成了仿菜谱精灵微信小程序的底部标签导航配置,单击不同的导航,可以切换显示不同的页面,同时导航图标和导航文字会呈现为选中状态,如图2.30所示。

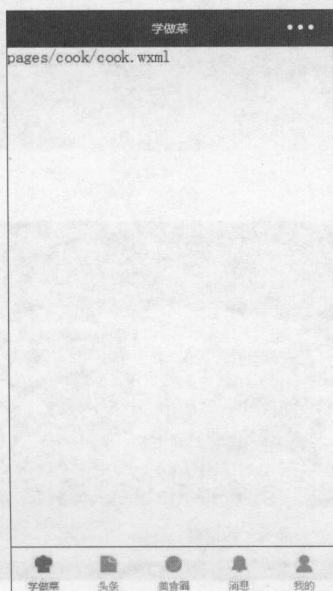


图2.30 “学做菜”界面

2.9.2 宫格导航设计

“学做菜”界面中海报轮播的图片,在微信小程序里有专门的swiper滑块视图组件来实现这个效果,在以后的章节中我们会介绍它的使用。宫格导航有菜谱分类、视频、美食养生、闪购4个导航,如图2.31所示。



图2.31 海报图片和宫格导航

1 进入pages/cook/cook.wxml文件，先设计海报轮播区域，采用一张图片来进行布局，将图片的宽度设置为100%，高度设置为230px，具体代码如下。

```
<view class="content">
  <view class="img">
    <image src="../../images/haibao/haibao-1.jpg" style="width:100%;height:230px;"></image>
  </view>
</view>
```

界面效果如图2.32所示。

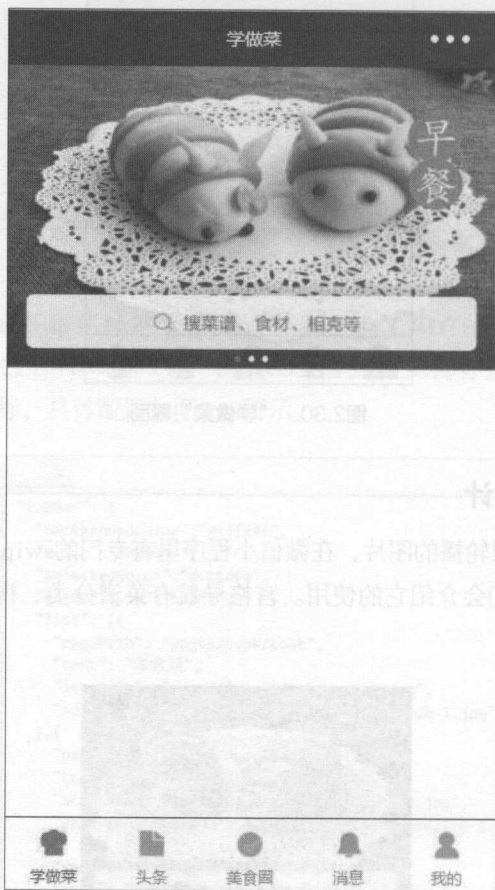


图2.32 海报图片

2 宫格导航分为4个导航：菜谱分类、视频、美食养生、闪购，每个导航对应一个图标，在导航的下面是灰色的间隔线，具体代码如下。

```
<view class="content">
  <view class="img">
    <image src="../../images/haibao/haibao-1.jpg" style="width:100%;height:230px;"></image>
  </view>

  <view class="nav">
    <view class="nav-item">
      <view> <image src="../../images/icon/fenlei.jpg" style="width:25px;height:23px;">
```

```

</image></view>
    <view> 菜谱分类 </view>
  </view>
  <view class="nav-item">
    <view> <image src="../../images/icon/shipin.jpg" style="width:25px;height:23px;">
</image></view>
    <view> 视频 </view>
  </view>
  <view class="nav-item">
    <view> <image src="../../images/icon/meishi.jpg" style="width:25px;height:23px;">
</image></view>
    <view> 美食养生 </view>
  </view>
  <view class="nav-item">
    <view> <image src="../../images/icon/shangou.jpg" style="width:25px;height:23px;">
</image></view>
    <view> 闪购 </view>
  </view>
  <view class="hr"></view>
</view>

```

3 进入pages/cook/cook.wxss文件，针对宫格导航添加样式，具体代码如下。

```

.nav{
  display: flex;
  flex-direction: row;
  text-align: center;
}
.nav-item{
  width: 25%;
  margin-top: 20px;
  font-size: 12px;
}
.hr{
  height: 15px;
  background-color: #cccccc;
  margin-top: 15px;
  opacity: 0.2;
}
.head{
  display: flex;
  flex-direction: row;
  margin: 10px;
  font-size: 13px;
  color: #999999;
}
.right{
  position: absolute;
  right: 10px;
  color: #cccccc;
}

```



```
.hr2{
  height: 2px;
  background-color: #cccccc;
  opacity: 0.2;
}
```

界面效果如图2.33所示。



图2.33 宫格导航

这样就完成了海报轮播区域和宫格导航区域的界面布局，在很多App上都会用海报轮播和宫格导航的方式来进行界面布局。

2.9.3 香哈头条初始化数据

微信小程序作为客户端，它的数据来源于服务端。下面模拟一下服务端提供的香哈头条列表的数据，有了数据，页面才能动态地进行渲染。

进入pages/cook/cook.js文件，添加initData函数，在data页面初始化数据里添加array数组，然后将initData定义的数据通过setData设置函数赋值给array数组，具体代码如下。

```
Page({
  data:{
    array:[]
  },
```

```
onLoad:function(options){
    var array = this.initData();
    this.setData({array:array});
},
initData:function(){
    var array = [];
    var object1 = new Object();
    object1.img = '../images/list/food-1.jpg';
    object1.title=' 爱心早餐 ';
    object1.type=' 健康养生 ';
    object1.liulan='20696 浏览 ';
    object1.pinglun='7 评论 ';
    array[0] = object1;

    var object2 = new Object();
    object2.img = '../images/list/food-2.jpg';
    object2.title=' 困了只想喝咖啡 ';
    object2.type=' 家庭医生在线 ';
    object2.liulan='29628 浏览 ';
    object2.pinglun='13 评论 ';
    array[1] = object2;

    var object3 = new Object();
    object3.img = '../images/list/food-3.jpg';
    object3.title=' 橘子吃多变小黄人 ';
    object3.type=' 家庭医生在线 ';
    object3.liulan='19585 浏览 ';
    object3.pinglun='6 评论 ';
    array[2] = object3;

    var object4 = new Object();
    object4.img = '../images/list/food-4.jpg';
    object4.title=' 搜狐新闻,手机用久了 ';
    object4.type=' 广告 ';
    object4.liulan='4688 浏览 ';
    object4.pinglun='4 评论 ';
    array[3] = object4;

    var object5 = new Object();
    object5.img = '../images/list/food-5.jpg';
    object5.title=' 困了只想喝咖啡 ';
    object5.type=' 家庭医生在线 ';
    object5.liulan='29628 浏览 ';
    object5.pinglun='13 评论 ';
    array[4] = object5;

    return array;
}
```

2.9.4 香哈头条列表渲染及绑定数据

香哈头条里有菜谱的图片、美食名称、分类、浏览数量以及评论数量，如图2.34所示。



图2.34 香哈头条列表

1 进入pages/cook/cook.wxml文件，进行香哈头条列表信息的界面布局，具体代码如下。

```
<view class="content">
  <view class="img">
    <image src="../../../images/haibao/haibao-1.jpg" style="width:100%;height:230px;"></image>
  </view>

  <view class="nav">
    <view class="nav-item">
      <view> <image src="../../../images/icon/fenlei.jpg" style="width:25px;height:23px;"></image>
      <view> 菜谱分类 </view>
    </view>
    <view class="nav-item">
      <view> <image src="../../../images/icon/shipin.jpg" style="width:25px;height:23px;"></image>
      <view> 视频 </view>
    </view>
```



```

    <view class="nav-item">
      <view> <image src="../../../images/icon/meishi.jpg" style="width:25px;height:23px;">
</image></view>
      <view> 美食养生 </view>
    </view>
    <view class="nav-item">
      <view> <image src="../../../images/icon/shangou.jpg" style="width:25px;height:23px;">
</image></view>
      <view> 闪购 </view>
    </view>
  </view>
  <view class="hr"></view>
  <view class="head">
    <view> 香哈头条 </view>
    <view class="right"></view>
  </view>
  <view class="list">
    <view class="item" bindtap="seeDetail" id="0">
      <view>
        <image src="../../../images/list/food-1.jpg" style="width:75px;height:58px;"></image>
      </view>
      <view class="desc">
        <view class="title"> 爱心早餐 </view>
        <view class="count">
          <view> 健康养生 </view>
          <view class="liulan">20696 浏览 </view>
          <view class="pinglun">7 评论 </view>
        </view>
      </view>
    </view>
  </view>
  <view class="hr2"></view>
</view>
</view>

```

2 进入pages/cook/cook.wxss文件，针对香哈头条列表信息添加样式，具体代码如下。

```

.item{
  display: flex;
  flex-direction: row;
  margin-left: 10px;
  margin-bottom: 5px;
}
.desc{
  margin-left: 20px;
  line-height: 30px;
}
.title{
  font-weight: bold;
}
.count{
  display: flex;
  flex-direction: row;

```

```
font-size: 12px;
color: #999999;
}
.liulan{
position: absolute;
right: 70px;
}
.pinglun{
position: absolute;
right: 10px;
}
```

界面效果如图2.35所示。



图2.35 香哈头条列表界面效果

3 现在香哈头条列表数据直接写在界面里，下面通过数据绑定和wx:for循环的方式动态加载数据，具体代码如下。

```
<view class="content">
  <view class="img">
    <image src="../images/haibao/haibao-1.jpg" style="width:100%;height:230px;"></image>
  </view>
```

```

<view class="nav">
  <view class="nav-item">
    <view> <image src="../../../images/icon/fenlei.jpg" style="width:25px;height:23px;">
</image></view>
    <view> 菜谱分类 </view>
  </view>
  <view class="nav-item">
    <view> <image src="../../../images/icon/shipin.jpg" style="width:25px;height:23px;">
</image></view>
    <view> 视频 </view>
  </view>
  <view class="nav-item">
    <view> <image src="../../../images/icon/meishi.jpg" style="width:25px;height:23px;">
</image></view>
    <view> 美食养生 </view>
  </view>
  <view class="nav-item">
    <view> <image src="../../../images/icon/shangou.jpg" style="width:25px;height:23
px;"></image></view>
    <view> 闪购 </view>
  </view>
</view>
<view class="hr"></view>
<view class="head">
  <view> 香哈头条 </view>
  <view class="right"></view>
</view>
<view class="list">
<block wx:for="{{array}}">
  <view class="item" bindtap="seeDetail" id="0">
    <view>
      <image src="{{item.img}}" style="width:75px;height:58px;"></image>
    </view>
    <view class="desc">
      <view class="title">{{item.title}}</view>
      <view class="count">
        <view>{{item.type}}</view>
        <view class="liulan">{{item.liulan}}</view>
        <view class="pinglun">{{item.pinglun}}</view>
      </view>
    </view>
  </view>
</block>
</view>
</view>

```

界面效果如图2.36所示。

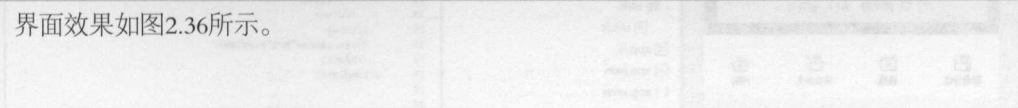


图2.36 界面效果如图2.36所示。



图2.36 香哈头条列表展现

在完成香哈头条列表展现时，先进行列表的界面布局，可以把数据写在页面上，页面布局完后，通过数据绑定的方式，将js里的data数据和WXML界面进行绑定，达到一种动态获取数据的效果。

2.9.5 香哈头条模板的引用

如果香哈头条列表信息在很多页面都是同样的展现方式，就可以把它制作成模板，达到一次制作、多次使用的效果。

1 在pages添加一个template目录，再添加一个template.wxml文件，在这个文件里制作一个香哈头条列表的模板，模板名称为cooks，将列表循环的内容放置在这个文件里，如图2.37所示。

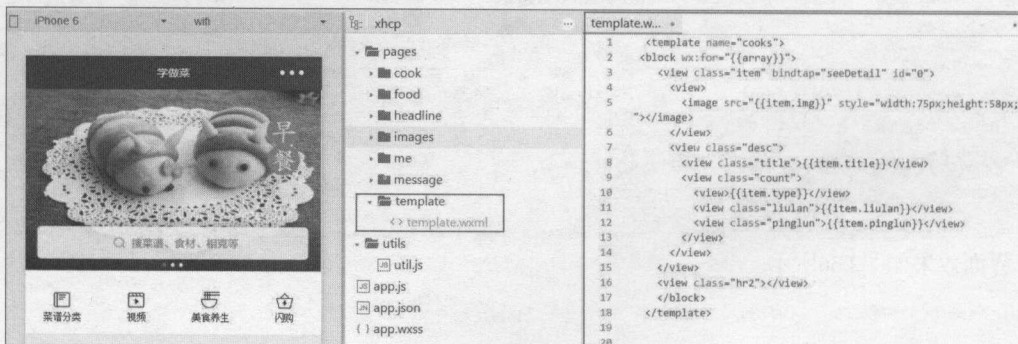


图2.37 香哈头条列表模板

2 将cooks模板引入到cook.wxml里，将列表信息展现出来，具体代码如下。

```
<view class="content">
  <view class="img">
    <image src="../../images/haibao/haibao-1.jpg" style="width:100%;height:230px;"></image>
  </view>

  <view class="nav">
    <view class="nav-item">
      <view> <image src="../../images/icon/fenlei.jpg" style="width:25px;height:23px;">
</image></view>
      <view> 菜谱分类 </view>
    </view>
    <view class="nav-item">
      <view> <image src="../../images/icon/shipin.jpg" style="width:25px;height:23px;">
</image></view>
      <view> 视频 </view>
    </view>
    <view class="nav-item">
      <view> <image src="../../images/icon/meishi.jpg" style="width:25px;height:23px;">
</image></view>
      <view> 美食养生 </view>
    </view>
    <view class="nav-item">
      <view> <image src="../../images/icon/shangou.jpg" style="width:25px;height:23px;">
</image></view>
      <view> 闪购 </view>
    </view>
  </view>
  <view class="hr"></view>
  <view class="head">
    <view> 香哈头条 </view>
    <view class="right"></view>
  </view>
  <import src="../../template/template" />
  <view class="list">
    <template is="cooks" data="{{array}}" />
  </view>
</view>
```

界面效果如图2.38所示。

可以看出，使用模板前后的界面效果是一样的。模板一般是在某个区域被不同的地方使用，像香哈头条列表一样，在其他地方也有列表的展现，这时就可以使用模板的方式，达到一次编写、多次引用的效果。



图2.38 香哈头条模板的使用

2.10 小结

微信小程序框架分析涉及很多内容，应重点掌握以下内容。

- 1** 了解微信小程序目录结构，理解框架全局文件、工具类文件、框架页面文件的使用。
- 2** 会配置窗口导航栏以及底部标签导航。
- 3** 了解微信小程序注册程序的应用及生命周期函数的意义和使用。
- 4** 掌握微信小程序注册页面，包括页面初始化数据、生命周期函数的使用、页面相关事件处理函数的使用、页面路由管理和setData设置函数的使用。
- 5** 学会微信小程序绑定数据。
- 6** 学会微信小程序条件判断和列表渲染的使用。
- 7** 学会微信小程序定义模板和引用功能。

第3章 用微信小程序组件构建UI界面

微信小程序框架里提供了很多UI组件，如视图容器组件、基础内容组件、丰富的表单组件、友好的操作反馈组件、页面链接的导航组件、视频音频播放的媒体组件、地图组件和画布组件。这些UI组件就像我们小时候玩的积木一样，使用这些积木可以建造一座房子、一座大桥，而每个UI组件也都有不同的用处，有了这些组件，就可以完成界面的布局和界面的渲染。

3.1 视图容器组件

视图容器组件共有3种：view视图容器、scroll-view可滚动视图区域、swiper滑块视图容器。

3.1.1 view视图容器

view视图容器是WXML界面布局的基础组件，它和HTML里的div功能类似，用来进行界面的布局。view视图容器也有自己的属性，如表3.1所示。

表3.1 view的属性

| 属性 | 类型 | 默认值 | 默认值 |
|------------------|---------|-------|---|
| hover | Boolean | false | 是否启用单击态 |
| hover-class | String | none | 指定按下去的样式类。当 hover-class= "none" 时，没有单击态效果 |
| hover-start-time | Number | 50 | 按住后多久出现单击态，单位为毫秒 |
| hover-stay-time | Number | 400 | 手指松开后单击态的保留时间，单位为毫秒 |

在WXML界面里使用view布局，渲染出界面内容，如图3.1所示。

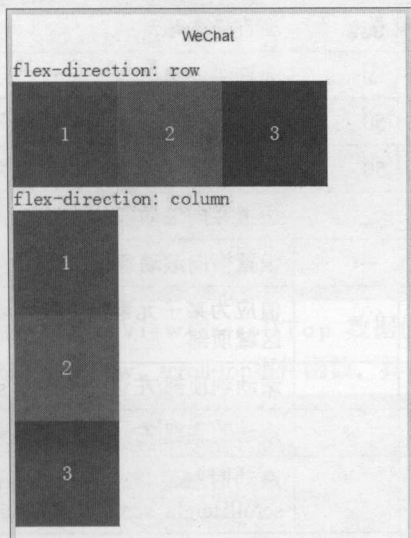


图3.1 view布局

具体代码如下。

```
<view class="section">
  <view class="section_title">flex-direction: row</view>
  <view class="flex-wrp" style="display: flex; flex-direction: row;">
    <view class="flex-item bc_green" style="width: 100px; height: 100px; background-color: green; color: #ffffff; text-align: center; line-height: 100px;">1</view>
    <view class="flex-item bc_red" style="width: 100px; height: 100px; background-color: red; color: #ffffff; text-align: center; line-height: 100px;">2</view>
    <view class="flex-item bc_blue" style="width: 100px; height: 100px; background-color: blue; color: #ffffff; text-align: center; line-height: 100px;">3</view>
  </view>
</view>
<view class="section">
  <view class="section_title">flex-direction: column</view>
  <view class="flex-wrp" style="display: flex; height: 300px; flex-direction: column;">
    <view class="flex-item bc_green" style="width: 100px; height: 100px; background-color: green; color: #ffffff; text-align: center; line-height: 100px;">1</view>
    <view class="flex-item bc_red" style="width: 100px; height: 100px; background-color: red; color: #ffffff; text-align: center; line-height: 100px;">2</view>
    <view class="flex-item bc_blue" style="width: 100px; height: 100px; background-color: blue; color: #ffffff; text-align: center; line-height: 100px;">3</view>
  </view>
</view>
```

3.1.2 scroll-view可滚动视图区域

scroll-view可滚动视图区域允许视图区域内容横向滚动或者纵向滚动，类似于浏览器横向滚动条和垂直滚动条的使用。scroll-view拥有自己的属性和事件，如表3.2所示。

表3.2 scroll-view的属性

| 属性 | 类型 | 默认值 | 说明 |
|-------------------|-------------|-------|---|
| scroll-x | Boolean | false | 允许横向滚动 |
| scroll-y | Boolean | false | 允许纵向滚动 |
| upper-threshold | Number | 50 | 距顶部/左边多远时（单位为px），触发 scrolltoupper 事件 |
| lower-threshold | Number | 50 | 距底部/右边多远时（单位为px），触发 scrolltolower 事件 |
| scroll-top | Number | | 设置竖向滚动条的位置 |
| scroll-left | Number | | 设置横向滚动条的位置 |
| scroll-into-view | String | | 值应为某子元素id，则滚动到该元素，元素顶部对齐滚动区域顶部 |
| bindscrolltoupper | EventHandle | | 滚动到顶部/左边，会触发 scrolltoupper 事件 |
| bindscrolltolower | EventHandle | | 滚动到底部/右边，会触发 scrolltolower 事件 |
| bindscroll | EventHandle | | 滚动时触发，event.detail = {scrollLeft, scrollTop, scrollHeight, scrollWidth, deltaX, deltaY} |

1. 纵向滚动

允许内容纵向滚动，需要给`<scroll-view>`一个固定高度，可以绑定滚动到顶部/左边（`bindscrolltoupper`）、滚动到底部/右边（`bindscrolltolower`）、滚动时（`bindscroll`）触发的事件，也可以滚动到指定的id区域（`scroll-into-view`）。下面实现纵向滚动，如图3.2所示。

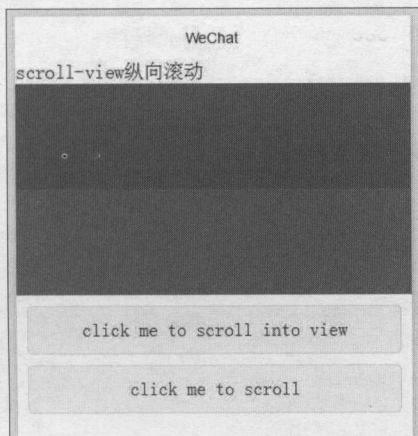


图3.2 纵向滚动

1 在WXML里使用`scroll-view`进行布局，设置`scroll-y="true"`纵向滚动，绑定`bindscrolltoupper`、`bindscrolltolower`、`bindscroll`、`scroll-into-view`、`scroll-top`事件，具体代码如下。

```
<view class="section">
  <view class="section__title">scroll-view 纵向滚动 </view>
  <scroll-view scroll-y="true" style="height: 200px;" bindscrolltoupper="upper"
    bindscrolltolower="lower" bindscroll="scroll" scroll-into-view="{{toView}}" scroll-
    top="{{scrollTop}}">
    <view id="green" style="width:100%;height:100px;background-color:green;"></view>
    <view id="red" style="width:100%;height:100px;background-color:red;"></view>
    <view id="yellow" style="width:100%;height:100px;background-color:yellow;"></view>
    <view id="blue" style="width:100%;height:100px;background-color:blue;"></view>
  </scroll-view>

  <view class="btn-area">
    <button type="default" style="margin:10px;" bindtap="tap">click me to scroll into
    view </button>
    <button type="default" style="margin:10px;" bindtap="tapMove">click me to
    scroll</button>
  </view>
```

2 在js里设置颜色的数组，绑定`toView`和`scrollTop` 数据值，提供`bindscrolltoupper`、`bindscrolltolower`、`bindscroll`、`scroll-into-view`、`scroll-top`事件函数，具体代码如下。

```
var order = ['red', 'yellow', 'blue', 'green', 'red']
Page({
  data: {
    toView: 'red',
    scrollTop: 100
  },
```



```

upper: function(e) {
  console.log(e)
},
lower: function(e) {
  console.log(e)
},
scroll: function(e) {
  console.log(e)
},
tap: function(e) {
  for (var i = 0; i < order.length; ++i) {
    if (order[i] === this.data.toView) {
      this.setData({
        toView: order[i + 1]
      })
      break
    }
  }
},
tapMove: function(e) {
  this.setData({
    scrollTop: this.data.scrollTop + 10
  })
}
})

```

这样就实现了纵向滚动，可以滚动到指定区域，也可以滚动到指定的位置，同时滚动顶部或底部会触发相应的事件，在滚动过程中也可以触发相应的事件。

2. 横向滚动

在使用今日头条或腾讯新闻时，在新闻列表的上方都会有新闻频道供我们单击，可以向左滑动和向右滑动来查看相应类别的新闻，可以采用scroll-view来实现这些新闻频道的横向滚动，如图3.3所示。



图3.3 新闻频道的横向滚动

在WXML里使用scroll-view进行布局,设置scroll-x="true" 横向滚动,具体代码如下。

```
<view class="section">
  <view class="section__title"> 新闻频道横向滚动 </view>
  <scroll-view scroll-x="true" style="width: 100%;">
    <view style="display: flex; flex-direction: row">
      <view style="margin-right: 10px;"> 推荐 </view>
      <view style="margin-right: 10px;"> 热点 </view>
      <view style="margin-right: 10px;"> 视频 </view>
      <view style="margin-right: 10px;"> 北京 </view>
      <view style="margin-right: 10px;"> 社会 </view>
      <view style="margin-right: 10px;"> 娱乐 </view>
      <view style="margin-right: 10px;"> 问答 </view>
      <view style="margin-right: 10px;"> 图片 </view>
      <view style="margin-right: 10px;"> 科技 </view>
      <view style="margin-right: 10px;"> 汽车 </view>
      <view style="margin-right: 10px;"> 社会 </view>
      <view style="margin-right: 10px;"> 娱乐 </view>
      <view style="margin-right: 10px;"> 问答 </view>
      <view style="margin-right: 10px;"> 图片 </view>
      <view style="margin-right: 10px;"> 科技 </view>
      <view style="margin-right: 10px;"> 汽车 </view>
    </view>
  </scroll-view>
</view>
```

这样就可以实现横向滚动了,可以向左滑动和向右滑动。



注意:

- (1) 请勿在 scroll-view 中使用 textarea、map、canvas、video 组件。
- (2) scroll-into-view 的优先级高于 scroll-top。
- (3) 在滚动 scroll-view 时会阻止页面回弹,所以在 scroll-view 中滚动无法触发 onPullDownRefresh。
- (4) 若要使用下拉刷新,请使用页面的滚动,而不是 scroll-view,这样也能通过单击顶部状态栏回到页面顶部。

3.1.3 swiper滑块视图容器

swiper滑块视图容器用来在指定区域内切换内容的显示,常用于制作海报轮播效果和页签内容的切换效果。它的属性如表3.3所示。

表3.3 swiper的属性

| 属性 | 类型 | 默认值 | 说明 |
|----------------|---------|-------|-----------|
| indicator-dots | Boolean | false | 是否显示面板指示点 |
| autoplay | Boolean | false | 是否自动切换 |

续表

| 属性 | 类型 | 默认值 | 说明 |
|------------|-------------|-------|--|
| current | Number | 0 | 当前所在页面的 index |
| interval | Number | 5 000 | 自动切换时间间隔 |
| duration | Number | 500 | 滑动动画时长 |
| circular | Boolean | false | 是否采用衔接滑动 |
| bindchange | EventHandle | | current 改变时会触发 change 事件，event.detail = {current: current} |

1. 海报轮播效果

海报轮播效果常用来展示商品图片信息或者广告信息，是很多网站或者App软件都会采用的一种布局方式，如图3.4和图3.5所示。

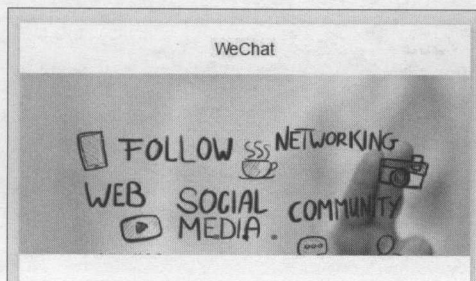


图3.4 海报1



图3.5 海报2

1 在WXML文件里进行海报轮播区域的布局，可采用swiper滑块视图容器组件进行布局，具体代码如下。

```
<view class="haibao">
  <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval="{{interval}}" duration="{{duration}}">
    <block wx:for="{{imgUrls}}">
      <swiper-item>
        <image src="{{item}}" class="slide-image" style="width:100%"></image>
      </swiper-item>
    </block>
  </swiper>
</view>
```

2 在js文件里，提供海报轮播的图片、是否自动播放、轮播的时长等数据，通过数据绑定的方式渲染到页面上，具体代码如下。

```
Page({
  data:{
    indicatorDots:true,
    autoplay:true,
    interval:5000,
    duration:1000,
    imgUrls:[
      "http://img06.tooopen.com/images/20160818/tooopen_sy_175866434296.jpg", "http://
```



```
img06.tooopen.com/images/20160818/tooopen_sy_175833047715.jpg", "http://img02.tooopen.com/images/20150928/tooopen_sy_143912755726.jpg"
    ]
  }
})
```

设置autoplay等于true时就可以自动进行海报轮播, 设置indicatorDots等于true时代表面板显示指示点, 同时可以设置interval自动切换时长、duration滑动动画时长。

2. 页签切换效果

swiper滑块视图容器除了可以用来进行海报轮播效果的实现外, 还可以进行页签切换效果的实现, 常用于多种方式的登录或者多种类别的切换, 如图3.6所示。

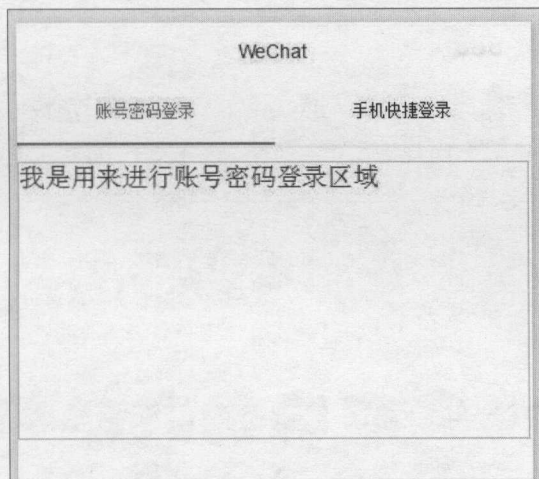


图3.6 页签切换效果

1 进入WXML文件, 进行账号密码登录和手机快捷登录的界面布局设计, 具体代码如下。

```
<view class="content">
  <view class="loginTitle">
    <view class="{{currentTab==0?'select':'default'}}" data-current="0" bindtap=
"switchNav"> 账号密码登录 </view>
    <view class="{{currentTab==1?'select':'default'}}" data-current="1" bindtap=
"switchNav"> 手机快捷登录 </view>
  </view>
  <view class="hr"></view>
  <swiper current="{{currentTab}}" style="height:{{winHeight}}px">
    <swiper-item>
      <view style="margin-top:10px;border:1px solid #cccccc;width:99%;height:200px;">
        我是用来进行账号密码登录区域
      </view>
    </swiper-item>
    <swiper-item>
      <view style="margin-top:10px;border:1px solid #cccccc;width:99%;height:200px;">
        我是用来进行手机快捷登录区域
      </view>
    </swiper-item>
  </swiper>
</view>
```

```

    </swiper-item>
  </swiper>
</view>

```

2 进入WXSS文件，给页面文件添加样式，具体代码如下。

```

.loginTitle{
  display: flex;
  flex-direction: row;
  width: 100%;
}
.select{
  font-size:12px;
  color: red;
  width: 50%;
  text-align: center;
  height: 45px;
  line-height: 45px;
  border-bottom:5rpx solid red;
}
.default{
  font-size:12px;
  margin: 0 auto;
  padding: 15px;
}
.hr{
  border: 1px solid #cccccc;
  opacity: 0.2;
}

```

3 进入js文件，提供窗口的宽度、高度、当前面板的索引值及页签切换函数，具体代码如下。

```

Page({
  data:{
    currentTab:0,
    winWidth:0,
    winHeight:0
  },
  onLoad:function(options){
    var page = this;
    wx.getSystemInfo({
      success: function(res) {
        console.log(res);
        page.setData({winWidth:res.windowWidth});
        page.setData({winHeight:res.windowHeight});
      }
    })
  },
  switchNav:function(e){
    var page = this;
    if(this.data.currentTab == e.target.dataset.current){
      return false;
    }else{

```

```
page.setData({currentTab:e.target.dataset.current});
}
}
})
```

这样就可以实现在两种登录状态下的页签切换效果。页签切换时，页签的标题呈现为选中的状态，同时对应的内容也跟着进行切换。

3.2 基础内容组件

基础内容组件包括icon图标组件、text文本组件、progress进度条组件。

3.2.1 icon图标组件

微信小程序提供了丰富的图标组件，应用于不同的场景，有成功、警告、提示、取消、下载等不同含义的图标，如图3.7所示。

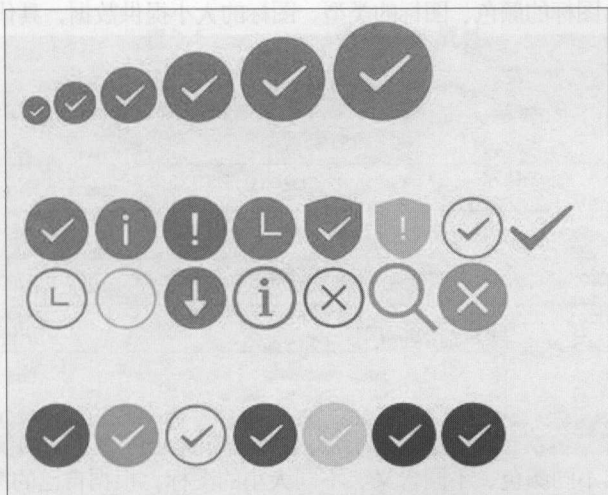


图3.7 图标

icon图标组件有3个属性：图标的类型type、图标的大小size和图标的颜色color，如表3.4所示。

表3.4 图标的属性

| 属性 | 类型 | 默认值 | 说明 |
|-------|--------|-----|--|
| type | String | | icon的类型，有效值：success, success_no_circle, info, warn, waiting, cancel, download, search, clear |
| size | Number | 23 | icon的大小，单位为px |
| color | Color | | icon的颜色，同css的color |

绘制出如图3.7所示图标的方法如下。

1 在WXML文件里，利用icon组件进行界面布局，具体代码如下。


```
<view class="group">
  <block wx:for="{{iconSize}}">
    <icon type="success" size="{{item}}" />
  </block>
</view>

<view class="group">
  <block wx:for="{{iconType}}">
    <icon type="{{item}}" size="45" />
  </block>
</view>

<view class="group">
  <block wx:for="{{iconColor}}">
    <icon type="success" size="45" color="{{item}}" />
  </block>
</view>
```

2 在js文件里，给图标的颜色、图标的类型、图标的大小提供数据，具体代码如下。

```
Page({
  data: {
    iconSize: [20, 30, 40, 50, 60, 70],
    iconColor: [
      'red', 'orange', 'yellow', 'green', 'rgb(0,255,255)', 'blue', 'purple'
    ],
    iconType: [
      'success', 'info', 'warn', 'waiting', 'safe_success', 'safe_warn',
      'success_circle', 'success_no_circle', 'waiting_circle', 'circle', 'download',
      'info_circle', 'cancel', 'search', 'clear'
    ]
  }
})
```

这样就可以绘制出不同颜色、不同含义、不同大小的图标，根据自己的需求，利用icon组件来设计自己的图标。

3.2.2 text文本组件

text文本组件支持转义符“\”，如换行\n、空格\t，<text/> 组件内只支持 <text/> 嵌套，除了文本节点，其他节点都无法长按选中。

下面，我们来看看转义符的使用，具体代码如下。

```
<view class="btn-area">
  <view class="body-view">
    <text>我爱北京 \t 我爱中国 </text>
    <text>我爱北京 \n 我爱中国 </text>
  </view>
</view>
```

界面效果如图3.8所示。

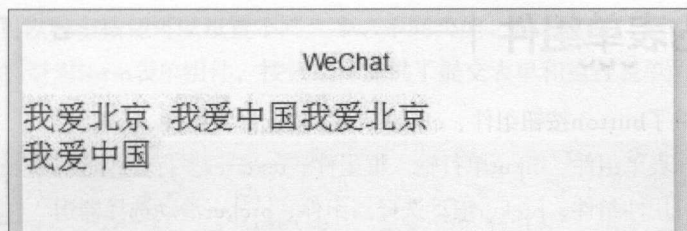


图3.8 转义符的效果

从图3.8中可以看出，\t具有空格功能，\n具有换行功能；同时也可以看出，text文本组件是放置在一行里，不同于view组件，每个view组件一行。

3.2.3 progress进度条组件

progress进度条组件是一种提高用户体验度的组件，就像视频播放一样，可以通过进度条看到完整视频的长度、当前播放的进度，这样能够让用户合理地安排自己的时间，提高用户的体验度。微信小程序也提供了progress进度条组件，它的属性如表3.5所示。

表3.5 progress进度条的属性

| 属性 | 类型 | 默认值 | 说明 |
|--------------|---------|---------|---------------|
| percent | Float | 无 | 百分比0~100 |
| show-info | Boolean | false | 在进度条右侧显示百分比 |
| stroke-width | Number | 6 | 进度条线的宽度，单位为px |
| color | Color | #09BB07 | 进度条的颜色 |
| active | Boolean | false | 进度条从左往右的动画 |

示例代码如下。

```
<progress percent="20" show-info />
<progress percent="40" stroke-width="12" />
<progress percent="60" color="pink" />
<progress percent="80" active />
```

界面效果参考图3.9。

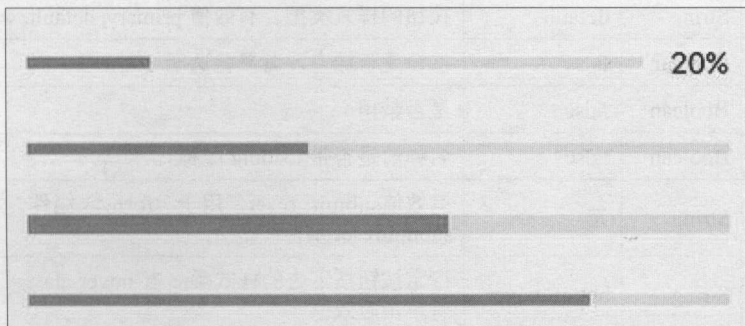


图3.9 进度条的效果

3.3 丰富的表单组件

微信小程序提供了button按钮组件、checkbox多项选择器组件、radio单项选择器组件、form表单组件、input单行输入框组件、textarea多行输入框组件、label改进表单可用性组件、picker滚动选择器组件、picker滑动选择器组件、switch开关选择器组件10种表单组件。

微课视频



表单UI组件

3.3.1 button按钮

button按钮组件提供3种类型的按钮：基本类型按钮、默认类型按钮和警告类型按钮，同时提供两种大小的按钮：默认和mini按钮，如图3.10所示。

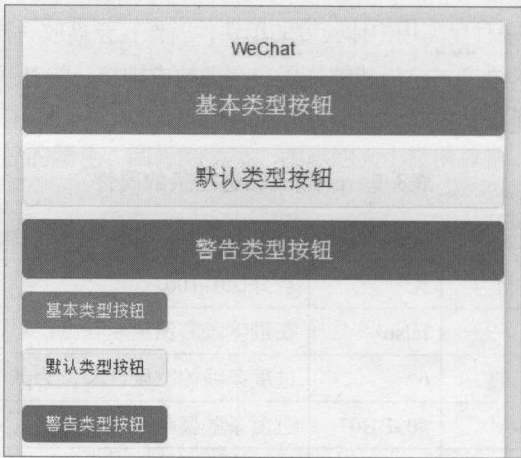


图3.10 按钮类型和大小

button按钮组件有很多属性，每个属性有不同的作用，如表3.6所示。

表3.6 button按钮的属性

| 属性 | 类型 | 默认值 | 说明 |
|------------------|---------|--------------|--|
| size | String | default | 有效值 default, mini |
| type | String | default | 按钮的样式类型，有效值 primary, default, warn |
| plain | Boolean | false | 按钮是否镂空，背景色透明 |
| disabled | Boolean | false | 是否禁用 |
| loading | Boolean | false | 名称前是否带 loading 图标 |
| form-type | String | 无 | 有效值submit, reset，用于 <form/> 组件，单击分别会触发 submit/reset 事件 |
| hover-class | String | button-hover | 指定按钮按下去的样式类。当 hover-class=" none" 时，没有单击态效果 |
| hover-start-time | Number | 50 | 按住后多久出现单击态，单位为毫秒 |
| hover-stay-time | Number | 400 | 手指松开后单击态的保留时间，单位为毫秒 |

从按钮属性中可以看出按钮可以设置不同大小、不同类型、是否镂空、是否禁用、按钮名称前是否带loading图标, 针对form表单组件, 按钮组件提供了提交表单和重置表单两个功能, 具体代码如下。

```
<button type="default" size="{{defaultSize}}" loading="{{loading}}" plain="{{plain}}"
  disabled="{{disabled}}" bindtap="default" > default </button>
<button type="primary" size="{{primarySize}}" loading="{{loading}}" plain="{{plain}}"
  disabled="{{disabled}}" bindtap="primary"> primary </button>
<button type="warn" size="{{warnSize}}" loading="{{loading}}" plain="{{plain}}"
  disabled="{{disabled}}" bindtap="warn"> warn </button>
<button bindtap="setDisabled"> 点击设置以上按钮 disabled 属性 </button>
<button bindtap="setPlain"> 点击设置以上按钮 plain 属性 </button>
<button bindtap="setLoading"> 点击设置以上按钮 loading 属性 </button>
```

```
var types = ['default', 'primary', 'warn']
var pageObject = {
  data: {
    defaultSize: 'default',
    primarySize: 'default',
    warnSize: 'default',
    disabled: false,
    plain: false,
    loading: false
  },
  setDisabled: function(e) {
    this.setData({
      disabled: !this.data.disabled
    })
  },
  setPlain: function(e) {
    this.setData({
      plain: !this.data.plain
    })
  },
  setLoading: function(e) {
    this.setData({
      loading: !this.data.loading
    })
  }
}

for (var i = 0; i < types.length; ++i) {
  (function(type) {
    pageObject[type] = function(e) {
      var key = type + 'Size'
      var changedData = {}
      changedData[key] =
        this.data[key] === 'default' ? 'mini' : 'default'
      this.setData(changedData)
    }
  })(types[i])
}
```

```
    }) (types[i])
  }
  Page(pageObject)
```

界面效果如图3.11所示。



图3.11 按钮效果

3.3.2 checkbox多项选择器

checkbox多项选择器组件，也就是我们常说的复选框，用来进行多项选择的时候会用到checkbox多项选择器。它的属性如表3.7所示。

表3.7 checkbox多项选择器的属性

| 属性 | 类型 | 默认值 | 说明 |
|----------|---------|-------|---|
| value | String | | <checkbox/>标识，选中时触发<checkbox-group/>的 change 事件，并携带 <checkbox/> 的 value |
| disabled | Boolean | false | 是否禁用 |
| checked | Boolean | false | 当前是否选中，可用来设置默认选中 |
| color | Color | | checkbox的颜色，同css的color |

checkbox-group是用来容纳多个checkbox多项选择器的容器，它有一个绑定事件bindchange，<checkbox-group/>中选中项发生改变时触发 change 事件，detail = {value:[选中的checkbox的value的数组]}。

下面，我们来演示一下checkbox多项选择器的使用，以及获取选中的value值。

1 在WXML文件里使用checkbox进行界面布局，具体代码如下。

```
<checkbox-group bindchange="checkboxChange">
  <checkbox value="USA" /> 美国
  <checkbox value="CHN" checked="true" /> 中国
</checkbox-group>
```

```
<checkbox value="BRA"/> 巴西
<checkbox value="JPN"/> 日本
<checkbox value="ENG" disabled/> 英国
</checkbox-group>
```

2 在js里，添加checkboxChange 事件函数，获取复选框选中的值，并将其打印出来，具体代码如下。

```
Page({
  checkboxChange:function(e){
    console.log(e.detail.value)
  }
})
```

界面效果如图3.12所示。

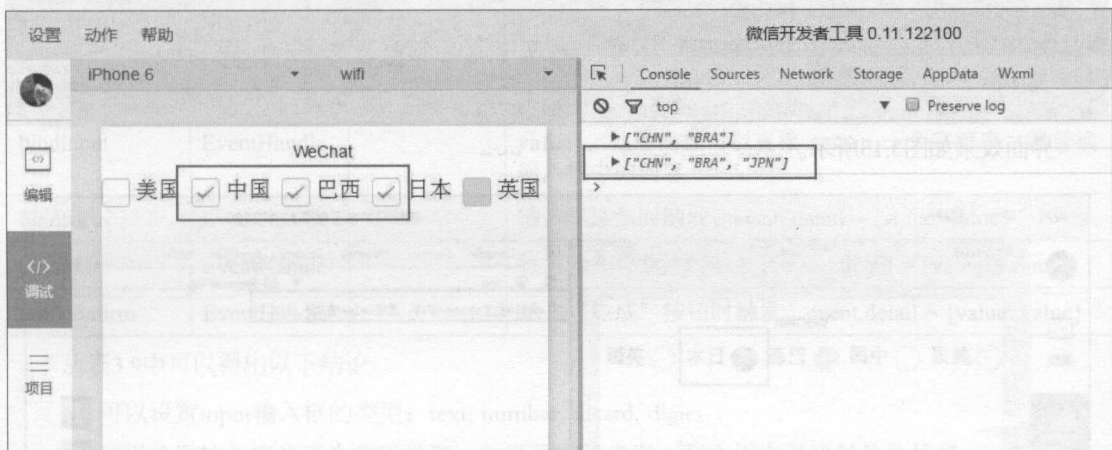


图3.12 复选框value值

从图3.12中可以看出，被禁用的复选框是不能进行使用的，在checkbox-group上面绑定bindchange事件，每次勾选时，会将所有勾选的复选框的值以数组的形式存在detail里。

3.3.3 radio单项选择器

radio单项选择器是与checkbox多项选择器对立的一个组件，它每次只能选中一个，选项间是一种互斥关系。它的属性如表3.8所示。

表3.8 radio单项选择器的属性

| 属性 | 类型 | 默认值 | 说明 |
|----------|---------|-------|--|
| value | String | | <radio/> 标识。当该<radio/> 选中时，<radio-group/> 的 change 事件会携带<radio/>的value |
| disabled | Boolean | false | 是否禁用 |
| checked | Boolean | false | 当前是否选中，可用来设置默认选中 |
| color | Color | | radio的颜色，同css的color |

radio-group是用来容纳多个radio单项选择器的容器，它有一个绑定事件bindchange，<radio-

group/> 中的选中项发生变化时触发 bindchange事件，event.detail = {value: 选中项radio的value}。

下面，我们来演示一下radio单项选择器的使用。

1 在WXML文件里使用radio单项选择器进行界面布局，具体代码如下。

```
<radio-group class="radio-group" bindchange="radioChange">
  <radio value="USA" /> 美国
  <radio value="CHN" checked/> 中国
  <radio value="BRA" disabled/> 巴西
  <radio value="JPN" /> 日本
  <radio value="ENG" /> 英国
</radio-group>
```

2 在js里，添加radioChange 事件函数，获取单项选中的值，并将其打印出来，具体代码如下。

```
Page({
  radioChange: function(e) {
    console.log('radio发生change事件，携带value值为: ', e.detail.value)
  }
})
```

界面效果如图3.13所示。

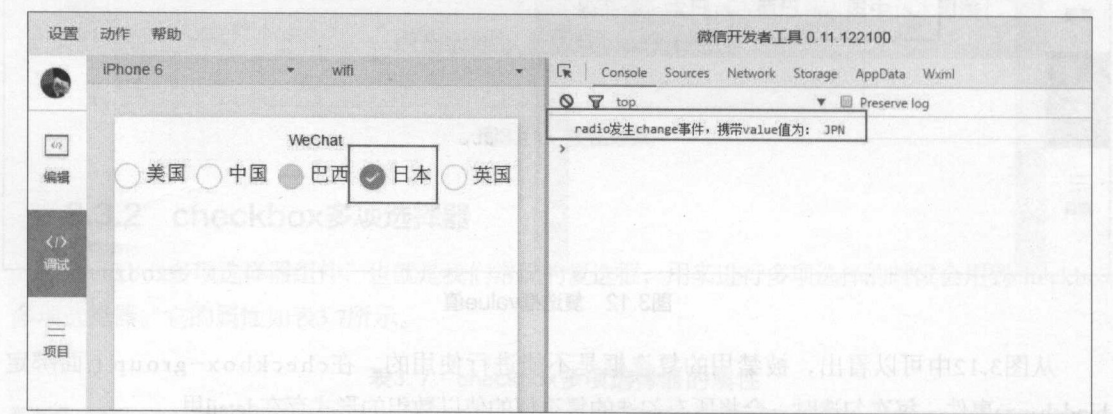


图3.13 单项选择器value值

从图3.13中可以看出，被禁用的单项选择器是不能进行使用的，在radio-group上面绑定 bindchange事件，每次勾选时，只能使一个选项呈现为选中状态，同时会将相应的值存在detail里。

3.3.4 input单行输入框

input单行输入框用来输入单行文本内容，其属性如表3.9所示。

表3.9 input单行输入框的属性

| 属性 | 类型 | 默认值 | 说明 |
|-------------|---------|-------|---|
| value | String | | 输入框的初始内容 |
| type | String | text | input 的类型，有效值：text, number, idcard, digit |
| password | Boolean | false | 是否是密码类型 |
| placeholder | String | | 输入框为空时的占位符 |

续表

| 属性 | 类型 | 默认值 | 说明 |
|-------------------|-------------|-------------------|---|
| placeholder-style | String | | 指定 placeholder 的样式 |
| placeholder-class | String | input-placeholder | 指定 placeholder 的样式类 |
| disabled | Boolean | false | 是否禁用 |
| maxlength | Number | 140 | 最大输入长度，设置为 -1 时不限制最大长度 |
| cursor-spacing | Number | 0 | 指定光标与键盘的距离，单位为px。取 input 距离底部的距离和 cursor-spacing 指定的距离的最小值作为光标与键盘的距离 |
| auto-focus | Boolean | false | (即将废弃，请直接使用 focus) 自动聚焦，拉起键盘 |
| focus | Boolean | false | 获取焦点 |
| bindinput | EventHandle | | 当键盘输入时，触发input事件，event.detail = {value: value}，处理函数可以直接 return 一个字符串，将替换输入框中的内容 |
| bindfocus | EventHandle | | 输入框聚焦时触发，event.detail = {value: value} |
| bindblur | EventHandle | | 输入框失去焦点时触发，event.detail = {value: value} |
| bindconfirm | EventHandle | | 单击“完成”按钮时触发，event.detail = {value: value} |

从表3.9中可以得出以下结论。

1 可以设置input输入框的类型：text, number, idcard, digit。

2 可以设置输入框是否为密码类型，如果是密码类型，则会用点号代替具体值显示。

3 通过placeholder来给输入框添加友好的提示信息，类似于“请输入手机号/用户名/邮箱”这样友好的输入框，placeholder-style设置提示信息的样式，placeholder-class设置提示信息的class，然后再针对这个class添加样式。

4 可以设置input输入框禁用和最大长度、获取焦点。

5 input输入框有3个常用的事件：输入时（bindinput）、光标聚焦时（bindfocus）、光标离开时（bindblur）。

示例代码如下。

1 在WXML里利用input单行输入框进行布局，具体代码如下。

```
<view class="section">
  <input placeholder=" 这是一个可以自动聚焦的 input" auto-focus/>
</view>
<view class="section">
  <input placeholder=" 这个只有在单击按钮的时候才聚焦 " focus="{{focus}}" />
  <view class="btn-area">
    <button bindtap="bindButtonTap">使得输入框获取焦点</button>
  </view>
</view>
<view class="section">
```

```

<input maxlength="10" placeholder="最大输入长度 10" />
</view>
<view class="section">
  <view class="section__title">你输入的是: {{inputValue}}</view>
  <input bindinput="bindKeyInput" placeholder="输入同步到 view 中" />
</view>
<view class="section">
  <input bindinput="bindReplaceInput" placeholder="连续的两个 1 会变成 2" />
</view>
<view class="section">
  <input bindinput="bindHideKeyboard" placeholder="输入 123 自动收起键盘" />
</view>
<view class="section">
  <input password type="number" />
</view>
<view class="section">
  <input password type="text" />
</view>
<view class="section">
  <input type="digit" placeholder="带小数点的数字键盘" />
</view>
<view class="section">
  <input type="idcard" placeholder="身份证输入键盘" />
</view>
<view class="section">
  <input placeholder-style="color:red" placeholder="占位符字体是红色的" />
</view>

```

2 在js文件中给input单行输入框添加相应的事件并提供数据，具体代码如下。

```

Page({
  data: {
    focus: false,
    inputValue: ''
  },
  bindButtonTap: function() {
    this.setData({
      focus: true
    })
  },
  bindKeyInput: function(e) {
    this.setData({
      inputValue: e.detail.value
    })
  },
  bindReplaceInput: function(e) {
    var value = e.detail.value
    var pos = e.detail.cursor
    if(pos !== -1){
      // 光标在中间
      var left = e.detail.value.slice(0,pos)
      // 计算光标的位置
    }
  }
})

```



```
    pos = left.replace(/11/g, '2').length
  }

  // 直接返回对象，可以对输入进行过滤处理，同时可以控制光标的位置
  return {
    value: value.replace(/11/g, '2'),
    cursor: pos
  }

  // 或者直接返回字符串，光标在最后边
  //return value.replace(/11/g, '2'),
},
bindHideKeyboard: function(e) {
  if (e.detail.value === '123') {
    // 收起键盘
    wx.hideKeyboard()
  }
}
})
```

界面效果如图3.14所示。

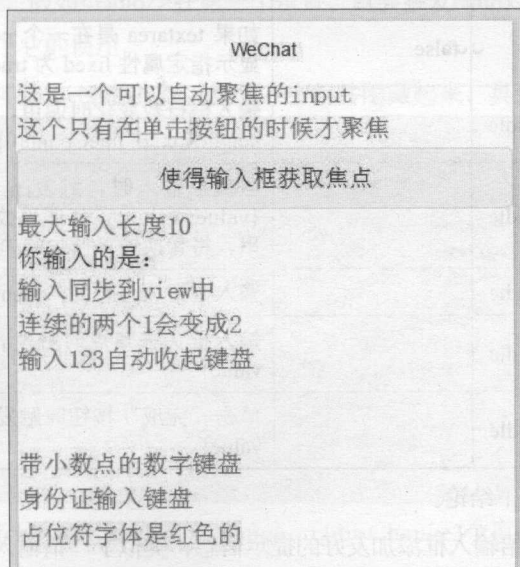


图3.14 input单行输入框



注意：input 组件是一个 native 组件，字体是系统字体，所以无法设置 font-family；在 input 聚焦期间，避免使用 css 动画。

3.3.5 textarea多行输入框

textarea多行输入框是与input单行输入框对应的一个组件，它可以输入多行文本内容。它的属性如表3.10所示。

表3.10 textarea多行输入框的属性

| 属性 | 类型 | 默认值 | 说明 |
|-------------------|-------------|-------------------|---|
| value | String | | 输入框的内容 |
| placeholder | String | | 输入框为空时的占位符 |
| placeholder-style | String | | 指定 placeholder 的样式 |
| placeholder-class | String | input-placeholder | 指定 placeholder 的样式类 |
| disabled | Boolean | false | 是否禁用 |
| maxlength | Number | 140 | 最大输入长度，设置为-1 时不限制最大长度 |
| cursor-spacing | Number | 0 | 指定光标与键盘的距离，单位为px。取 textarea 距离底部的距离和 cursor-spacing 指定的距离的最小值作为光标与键盘的距离 |
| auto-focus | Boolean | false | （即将废弃，请直接使用 focus）自动聚焦，拉起键盘 |
| focus | Boolean | false | 获取焦点 |
| auto-height | Boolean | false | 是否自动增高，设置auto-height时，style.height不生效 |
| fixed | Boolean | false | 如果 textarea 是在一个 position:fixed 的区域，需要显示指定属性 fixed 为 true |
| bindlinechange | EventHandle | | 输入框行数变化时调用，event.detail = {height: 0, heightRpx: 0, lineCount: 0} |
| bindinput | EventHandle | | 当键盘输入时，触发input事件，event.detail = {value: value}，处理函数可以直接 return 一个字符串，将替换输入框中的内容 |
| bindfocus | EventHandle | | 输入框聚焦时触发，event.detail = {value: value} |
| bindblur | EventHandle | | 输入框失去焦点时触发，event.detail = {value: value} |
| bindconfirm | EventHandle | | 单击“完成”按钮时触发，event.detail = {value: value} |

从表3.10中可以得出以下结论。

- 1

通过placeholder来给输入框添加友好的提示信息，类似于“请输入手机号/用户名/邮箱”这样友好的输入框，placeholder-style设置提示信息的样式，placeholder-class设置提示信息的class，然后再针对这个class添加样式。
- 2

可以设置textarea输入框禁用和最大长度、获取焦点、自动调整行高。
- 3

input输入框有 4 个常用的事件：输入时（bindinput）、光标聚焦时（bindfocus）、光标离开时（bindblur）、行数变化时（bindlinechange）。
- 示例代码如下。

```
<view class="section">
  <textarea bindblur="bindTextAreaBlur" auto-height placeholder="自动变高" />
</view>
<view class="section">
```

```
<textarea placeholder="placeholder 颜色是红色的 " placeholder-style="color:red;" />
</view>
```



注意：请勿在 scroll-view 中使用 textarea 组件，textarea 的 blur 事件会晚于页面上的 tap 事件，如果需要在 button 的单击事件中获取 textarea，可以使用 form 的 bindsubmit，不建议在多行文本上对用户的输入进行修改，所以 textarea 的 bindinput 处理函数并不会将返回值反映到 textarea 上。

3.3.6 label组件

label组件用来改进表单的可用性，目前可以用来改进的组件有<button/>、<checkbox/>、<radio/>、<switch/>。它只有一个属性for，用来绑定控件的id。它的使用有两种方式：一种是定义for属性，另一种是没有定义for属性。

1. label组件没有定义for属性

label组件没有定义for属性时，在label内包含<button/>、<checkbox/>、<radio/>、<switch/>这些组件，当单击label组件时，会触发label内包含的第一个控件，假如<button/>在第一个位置，就会触发<button/>对应的事件，假如<radio/>在第一个位置，就会触发radio对应的事件。

下面，我们来演示一下它的使用。

1 在WXML文件里利用label组件布局，把第一个组件隐藏起来，具体代码如下。

```
<label>
  <button bindtap="clickBtn" hidden>我是 button 按钮 </button>
  <view>我是 label 组件内的内容 </view>
  <checkbox-group bindchange="checkboxChange">
    <checkbox value=" 中国 " /> 中国
    <checkbox value=" 美国 " /> 美国
  </checkbox-group>
  <radio-group bindchange="radioChange">
    <radio value=" 男 " /> 男
    <radio value=" 女 " /> 女
  </radio-group>
</label>
```

2 在js文件里添加clickBtn、checkboxChange、radioChange3个事件函数，分别打印不同的信息，具体代码如下。

```
Page({
  clickBtn:function(){
    console.log(" 单击了按钮组件 ");
  },
  checkboxChange:function(){
    console.log(" 单击了多项选择器组件 ");
  },
  radioChange:function(){
    console.log(" 单击了单项选择器组件 ");
  }
})
```


3 在WXML界面里可以看到<button/>按钮组件是隐藏起来的，但是单击“我是label组件内的内容”，可以看到打印信息是按钮事件函数打印的信息，如图3.15所示。

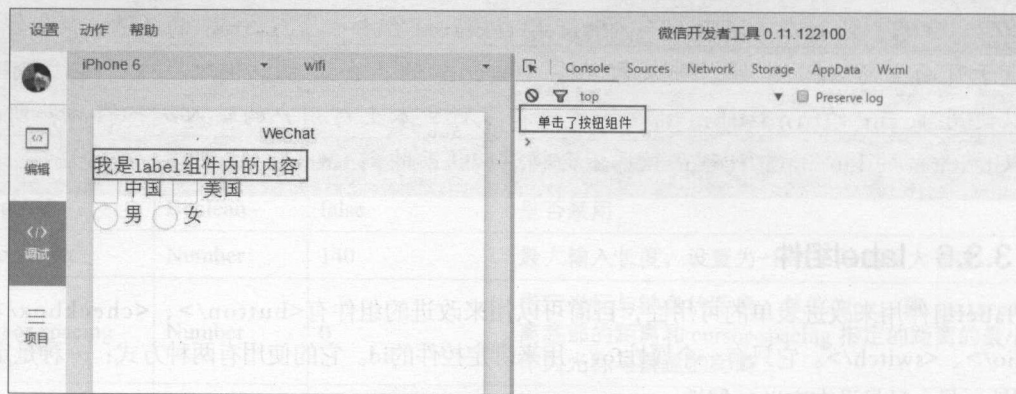


图3.15 没有定义for属性

从这里可以看出，label组件内有多个组件时，会触发第一个组件。

2. label组件定义for属性

label组件定义for属性后，会根据for属性的值找到组件id一样的值，然后会触发这个组件的相应事件。

下面，我们来演示一下它的使用。

1 在WXML文件里利用label组件布局，把第一个组件隐藏起来，给label定义for等于man，让它找到id值等于man组件，然后触发该组件的事件，具体代码如下。

```
<label for="man">
  <button id="btn" bindtap="clickBtn" hidden> 我是 button 按钮 </button>
  <view> 我是 label 组件内的内容 </view>
  <checkbox-group bindchange="checkboxChange" id="checkbox">
    <checkbox value=" 中国 " /> 中国
    <checkbox value=" 美国 " /> 美国
  </checkbox-group>
  <radio-group bindchange="radioChange" >
    <radio id="man" value=" 男 "/> 男
    <radio id="women" value=" 女 "/> 女
  </radio-group>
</label>
```

2 在js文件里添加clickBtn、checkboxChange、radioChange3个事件函数，分别打印不同的信息，具体代码如下。

```
Page({
  clickBtn:function(){
    console.log(" 单击了按钮组件 ");
  },
  checkboxChange:function(){
    console.log(" 单击了多项选择器组件 ");
  },
  radioChange:function(){
```

```

console.log("单击了单项选择器组件");
}
})

```

3 在WXML界面里可以看到<button/>按钮组件是隐藏起来的，但是单击“我是label组件内的内容”，可以看到id值等于man的单项选择器程序为选中状态，同时触发事件，打印信息，如图3.16所示。

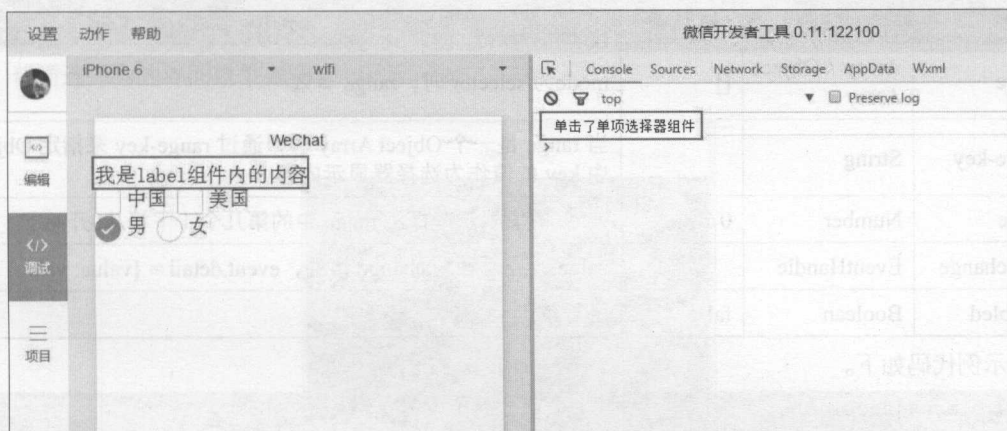


图3.16 定义for属性

从这里可以看出，如果label定义for属性，就会根据for属性的值找到组件id等于for属性值，然后触发相应事件，如果label没有定义for属性，它会找到label组件内的第一个组件，然后触发相应事件。

3.3.7 picker滚动选择器

picker滚动选择器有3种：普通选择器、时间选择器和日期选择器，默认是普通选择器，如图3.17~图3.19所示。

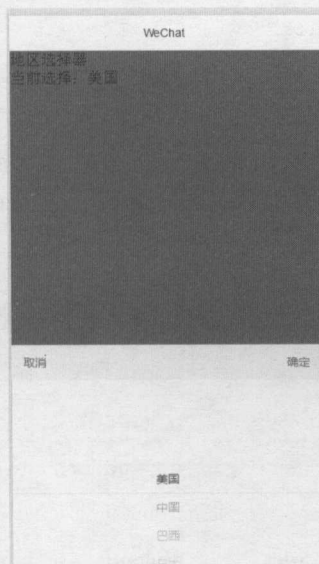


图3.17 普通选择器

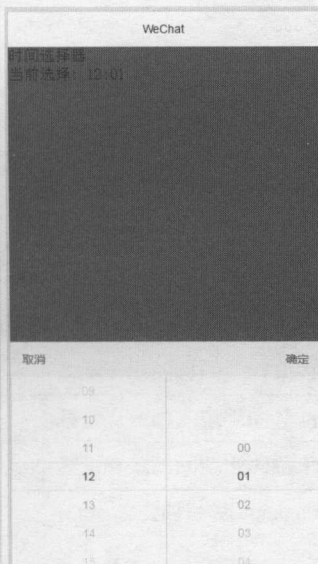


图3.18 时间选择器

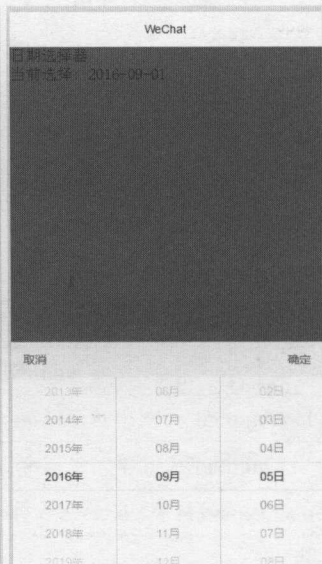


图3.19 日期选择器

这3种选择器是通过mode来区分的，普通选择器mode = selector，时间选择器mode = time，日期选择器mode = date，每种类型的选择器的属性不同，如表3.11～表3.13所示。

1. 普通选择器：mode = selector

普通选择器的属性如表3.11所示。

表3.11 普通选择器的属性

| 属性 | 类型 | 默认值 | 说明 |
|------------|-------------------------|-------|--|
| range | Array / Object Array | [] | mode为 selector 时，range 有效 |
| range-key | String | | 当 range 是一个 Object Array 时，通过 range-key 来指定 Object 中 key 的值作为选择器显示内容 |
| value | Number | 0 | value 的值表示选择了 range 中的第几个（下标从0开始） |
| bindchange | EventHandle | | value 改变时触发 change 事件，event.detail = {value: value} |
| disabled | Boolean | false | 是否禁用 |

示例代码如下。

```
<view class="section">
  <view class="section__title">地区选择器</view>
  <picker bindchange="bindPickerChange" value="{{index}}" range="{{array}}">
    <view class="picker">
      当前选择: {{array[index]}}
    </view>
  </picker>
</view>
```

```
Page({
  data: {
    array: ['美国', '中国', '巴西', '日本'],
    objectArray: [
      {
        id: 0,
        name: '美国'
      },
      {
        id: 1,
        name: '中国'
      },
      {
        id: 2,
        name: '巴西'
      },
      {
        id: 3,
        name: '日本'
      }
    ]
  },
  bindPickerChange(e) {
    console.log('picker onchange', e.detail.value)
  }
})
```



```
index: 0
},
bindPickerChange: function(e) {
  console.log('picker 发送选择改变，携带值为 ', e.detail.value)
  this.setData({
    index: e.detail.value
  })
}
})
```

普通选择器的界面效果如图3.20所示。

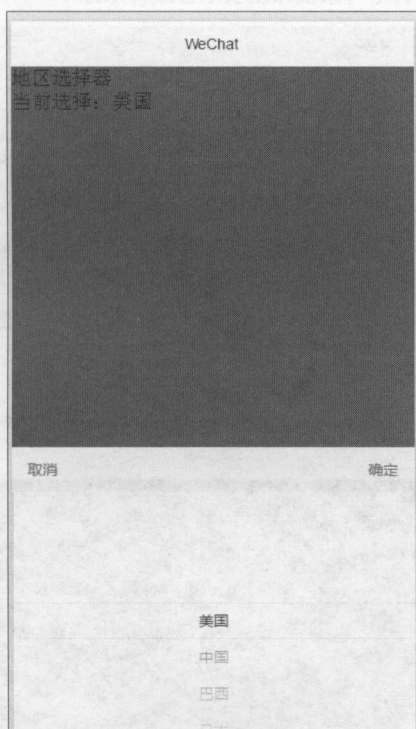


图3.20 普通选择器

2. 时间选择器：mode = time

时间选择器的属性如表3.12所示。

表3.12 时间选择器的属性

| 属性 | 类型 | 默认值 | 说明 |
|------------|-------------|-------|---|
| value | String | | 表示选中的时间，格式为"hh:mm" |
| start | String | | 表示有效时间范围的开始，字符串的格式为"hh:mm" |
| end | String | | 表示有效时间范围的结束，字符串的格式为"hh:mm" |
| bindchange | EventHandle | | value 改变时触发 change 事件，event.detail = {value: value} |
| disabled | Boolean | false | 是否禁用 |

示例代码如下。

```
<view class="section">
  <view class="section__title"> 时间选择器 </view>
  <picker mode="time" value="{{time}}" start="09:01" end="21:01" bindchange=
    "bindTimeChange">
    <view class="picker">
      当前选择： {{time}}
    </view>
  </picker>
</view>
```

```
Page({
  data: {
    time: '12:01'
  },
  bindTimeChange: function(e) {
    this.setData({
      time: e.detail.value
    })
  }
})
```

时间选择器的界面效果如图3.21所示。

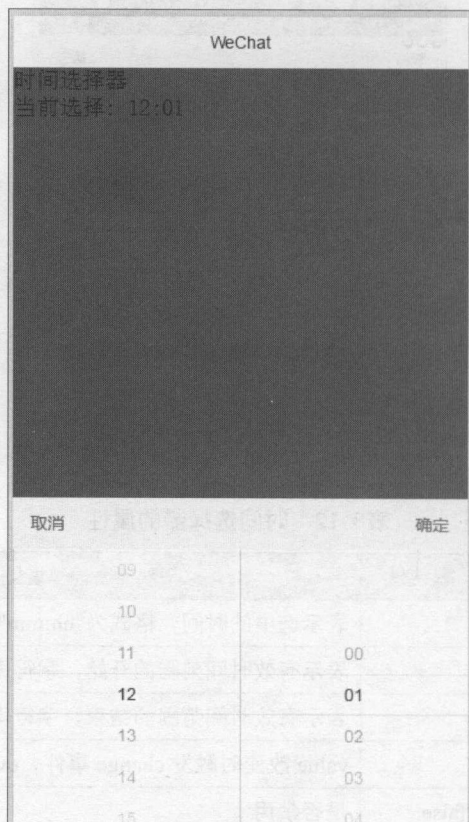


图3.21 时间选择器

3. 日期选择器：mode = date

日期选择器的属性如表3.13所示。

表3.13 日期选择器的属性

| 属性 | 类型 | 默认值 | 说明 |
|------------|-------------|-------|---|
| value | String | 0 | 表示选中的日期，格式为"YYYY-MM-DD" |
| start | String | | 表示有效日期范围的开始，字符串的格式为"YYYY-MM-DD" |
| end | String | | 表示有效日期范围的结束，字符串的格式为"YYYY-MM-DD" |
| fields | String | day | 有效值 year,month,day，表示选择器的粒度 |
| bindchange | EventHandle | | value 改变时触发 change 事件，event.detail = {value: value} |
| disabled | Boolean | false | 是否禁用 |

示例代码如下。

```
<view class="section">
  <view class="section__title">日期选择器 </view>
  <picker mode="date" value="{{date}}" start="2015-09-01" end="2017-09-01" bindchange
= "bindDateChange">
    <view class="picker">
      当前选择：{{date}}
    </view>
  </picker>
</view>
```

```
Page({
  data: {
    date: '2016-09-01'
  },
  bindDateChange: function(e) {
    this.setData({
      date: e.detail.value
    })
  }
})
```

日期选择器的界面效果如图3.22所示。

4. picker-view嵌入页面滚动选择器

除了普通选择器、时间选择器和日期选择器3种滚动选择器之外，还有一种嵌入页面的滚动选择器，使用picker-view组件在页面里的布局，如图3.23所示。

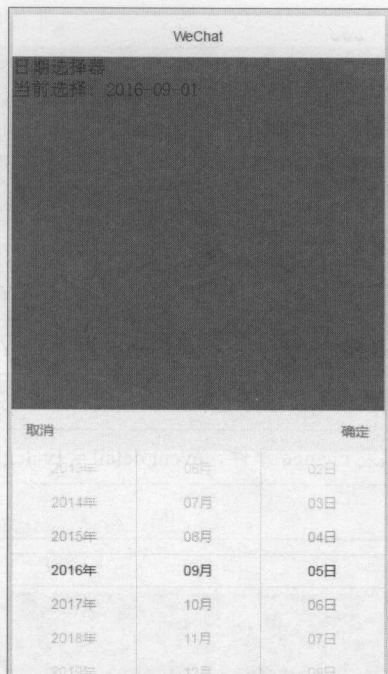


图3.22 日期选择器

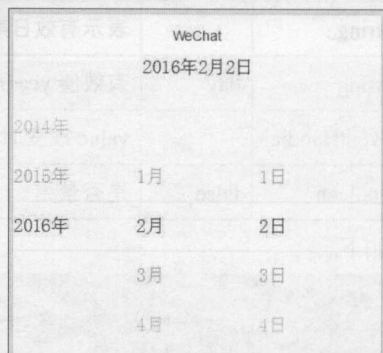


图3.23 picker-view嵌入页面滚动选择器

picker-view嵌入页面滚动选择器组件里面只有<picker-view-column/>组件，其他节点不会显示。picker-view有3个属性，如表3.14所示。

表3.14 picker-view嵌入页面滚动选择器的属性

| 属性 | 类型 | 默认值 | 说明 |
|-----------------|-------------|-------|--|
| value | Number | Array | 数组中的数字依次表示 picker-view 内的 picker-view-columne 选择的第几项（下标从 0 开始），数字大于 picker-view-column 可选项长度时，选择最后一项 |
| indicator-style | String | | 设置选择器中间选中框的样式 |
| bindchange | EventHandle | | 当滚动选择，value 改变时触发 change 事件，event.detail = {value: value}; value为数组，表示 picker-view 内的 picker-view-column 当前选择的是第几项（下标从 0 开始） |

示例代码如下。

```
<view>
  <view style="text-align:center">{{year}}年{{month}}月{{day}}日</view>
  <picker-view indicator-style="height: 50px;" style="width: 100%; height: 300px;"
value="{{value}}" bindchange="bindChange">
    <picker-view-column>
      <view wx:for="{{years}}" style="line-height: 50px">{{item}}年</view>
    </picker-view-column>
    <picker-view-column>
      <view wx:for="{{months}}" style="line-height: 50px">{{item}}月</view>
    </picker-view-column>
  </picker-view>
</view>
```

```

<picker-view-column>
  <view wx:for="{{days}}" style="line-height: 50px">{{item}} 日 </view>
</picker-view-column>
</picker-view>
</view>

```

```

const date = new Date()
const years = []
const months = []
const days = []

for (let i = 1990; i <= date.getFullYear(); i++) {
  years.push(i)
}

for (let i = 1 ; i <= 12; i++) {
  months.push(i)
}

for (let i = 1 ; i <= 31; i++) {
  days.push(i)
}

Page({
  data: {
    years: years,
    year: date.getFullYear(),
    months: months,
    month: 2,
    days: days,
    day: 2,
    year: date.getFullYear(),
    value: [9999, 1, 1],
  },
  bindChange: function(e) {
    const val = e.detail.value
    this.setData({
      year: this.data.years[val[0]],
      month: this.data.months[val[1]],
      day: this.data.days[val[2]]
    })
  }
})

```

3.3.8 slider滑动选择器

slider滑动选择器组件经常用来控制声音的大小、屏幕的亮度等场景的使用，它可以设置滑动步长、显示当前值以及设置最小/最大值，如图3.24所示。

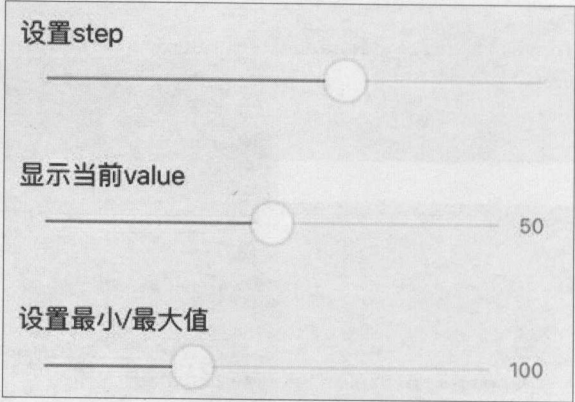


图3.24 slider滑动选择器

slider滑动选择器的属性如表3.15所示。

表3.15 slider滑动选择器的属性

| 属性 | 类型 | 默认值 | 说明 |
|----------------|-------------|---------|--|
| min | Number | 0 | 最小值 |
| max | Number | 100 | 最大值 |
| step | Number | 1 | 步长，取值必须大于 0，并且可被(max - min)整除 |
| disabled | Boolean | false | 是否禁用 |
| value | Number | 0 | 当前取值 |
| color | Color | #e9e9e9 | 背景条的颜色 |
| selected-color | Color | #1aad19 | 已选择的颜色 |
| show-value | Boolean | false | 是否显示当前 value |
| bindchange | EventHandle | | 完成一次拖动后触发的事件，event.detail = {value: value} |

示例代码如下。

```
<view class="section section_gap">
  <text class="section__title"> 设置 step</text>
  <view class="body-view">
    <slider bindchange="sliderchange" step="5"/>
  </view>
</view>

<view class="section section_gap">
  <text class="section__title"> 显示当前 value</text>
  <view class="body-view">
    <slider bindchange="sliderchange" show-value/>
  </view>
</view>
```



```
<view class="section section_gap">
  <text class="section__title">设置最小 / 最大值 </text>
  <view class="body-view">
    <slider bindchange="sliderchange" min="50" max="200" show-value/>
  </view>
</view>

<view class="section section_gap">
  <text class="section__title">设置颜色 </text>
  <view class="body-view">
    <slider bindchange="sliderchange" color="black" selected-color="red"/>
  </view>
</view>

<view class="section section_gap">
  <text class="section__title">禁用 </text>
  <view class="body-view">
    <slider bindchange="sliderchange" disabled show-value/>
  </view>
</view>
```

slider滑动选择器的界面效果如图3.25所示。

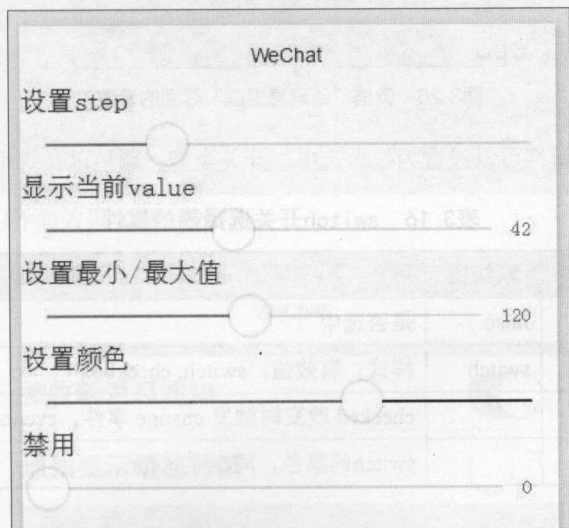


图3.25 slider滑动选择器

3.3.9 switch开关选择器

switch开关选择器应用得十分普遍，它有两个状态：开或者关，在很多场景都会用到开关这个功能，如微信设置里的“新消息提醒”界面，通过开关来设置是否接收消息、显示消息、是否有声音、是否震动等功能，如图3.26所示。

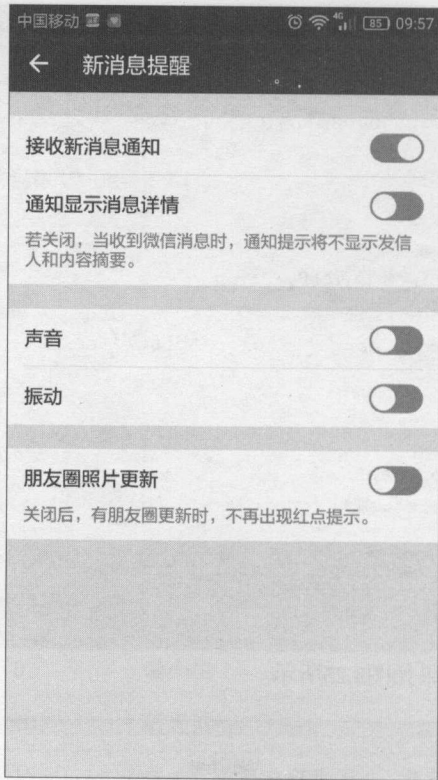


图3.26 微信“新消息提醒”界面的设置

switch开关选择器的属性可以设置为是否选中、开关类型、颜色以及绑定事件，如表3.16所示。

表3.16 switch开关选择器的属性

| 属性 | 类型 | 默认值 | 说明 |
|------------|-------------|--------|--|
| checked | Boolean | false | 是否选中 |
| type | String | switch | 样式，有效值：switch, checkbox |
| bindchange | EventHandle | | checked 改变时触发 change 事件，event.detail={value:checked} |
| color | Color | | switch 的颜色，同css的color |

示例代码如下。

```
<view style="background-color:#cccccc;height:600px;">
  <view style="padding-top:10px;"></view>
  <view style="display:flex;flex-direction:row;background-color:#ffffff;height:50px;
line-height:50px;">
    <view style="font-weight:bold;">接收新消息通知</view>
    <view style="position:absolute;right:10px;">
      <switch type="switch" checked/>
    </view>
  </view>
  <view style="height:1px;background-color:#f2f2f2;opacity:0.2"></view>
  <view style="display:flex;flex-direction:row;background-color:#ffffff;height:50px;
```

```

line-height:50px;">
  <view style="font-weight:bold;">通知显示消息详情 </view>
  <view style="position:absolute;right:10px;">
    <switch type="switch"/>
  </view>
</view>
<view style="height:1px;background-color:#f2f2f2;opacity:0.2"></view>

<view style="margin-top:20px;"></view>
<view style="height:1px;background-color:#f2f2f2;opacity:0.2"></view>
<view style="display:flex;flex-direction:row;background-color:#ffffff;height:50px;
line-height:50px;">
  <view style="font-weight:bold;">声音 </view>
  <view style="position:absolute;right:10px;">
    <switch type="checkbox" checked/>
  </view>
</view>
<view style="height:1px;background-color:#f2f2f2;opacity:0.2"></view>
<view style="height:1px;background-color:#f2f2f2;opacity:0.2"></view>
<view style="display:flex;flex-direction:row;background-color:#ffffff;height:50px;
line-height:50px;">
  <view style="font-weight:bold;">震动 </view>
  <view style="position:absolute;right:10px;">
    <switch type="checkbox"/>
  </view>
</view>
<view style="height:1px;background-color:#f2f2f2;opacity:0.2"></view>
</view>

```

switch开关选择器的界面效果如图3.27所示。

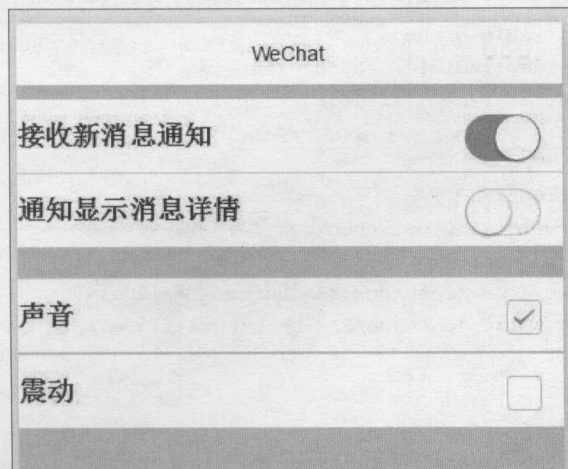


图3.27 switch开关选择器

3.3.10 form表单

form表单组件用来将表单里组件的值提交给Java Script逻辑层进行处理，它可以提交<switch/>

<input/> <checkbox/> <slider/> <radio/> <picker/>这些组件的值。提交表单的时候，会借助于button组件的formType为submit的属性，将表单组件中的 value 值进行提交，需要在表单组件中加上name 来作为 key。form表单的属性如表3.17所示。

表3.17 form表单的属性

| 属性 | 类型 | 默认值 | 说明 |
|---------------|-------------|-----|---|
| report-submit | Boolean | | 是否返回 formId 用于发送模板消息 |
| bindsubmit | EventHandle | | 携带 form 中的数据触发 submit 事件，event.detail = {value : { 'name' : 'value' }, formId: '' } |
| bindreset | EventHandle | | 表单重置时会触发 reset 事件 |

示例代码如下。

```
<form bindsubmit="formSubmit" bindreset="formReset">
  <view style="margin:10px;">
    <view style="font-weight:bold;">switch 开关选择器 </view>
    <switch name="switch"/>
  </view>
  <view style="margin:10px;">
    <view style="font-weight:bold;">slider 滑动选择器 </view>
    <slider name="slider" show-value ></slider>
  </view>

  <view style="margin:10px;">
    <view style="font-weight:bold;">input 单行输入框 </view>
    <input name="input" placeholder="please input here" />
  </view>
  <view style="margin:10px;">
    <view style="font-weight:bold;">radio 单项选择器 </view>
    <radio-group name="radio-group">
      <label><radio value="radio1"/>radio1</label>
      <label><radio value="radio2"/>radio2</label>
    </radio-group>
  </view>
  <view style="margin:10px;">
    <view style="font-weight:bold;">checkbox 多项选择器 </view>
    <checkbox-group name="checkbox">
      <label><checkbox value="checkbox1"/>checkbox1</label>
      <label><checkbox value="checkbox2"/>checkbox2</label>
    </checkbox-group>
  </view>
  <view class="btn-area">
    <button formType="submit" type="primary">Submit</button>
    <button formType="reset">Reset</button>
  </view>
</form>
```

```
formSubmit: function(e) {  
    console.log('form 发生了 submit 事件, 携带数据为: ', e.detail.value)  
},  
formReset: function() {  
    console.log('form 发生了 reset 事件')  
}  
})
```

界面效果如图3.28和图3.29所示。

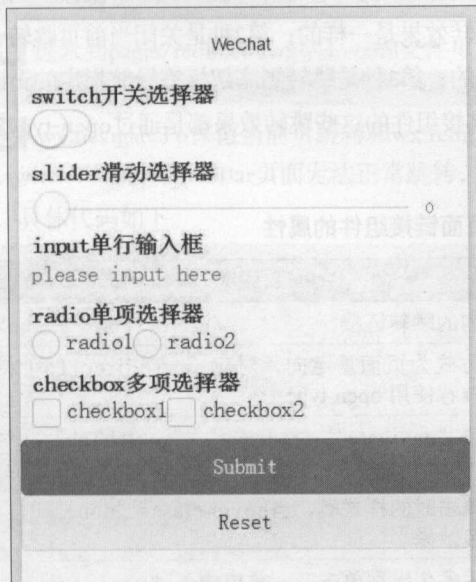


图3.28 未填写表单

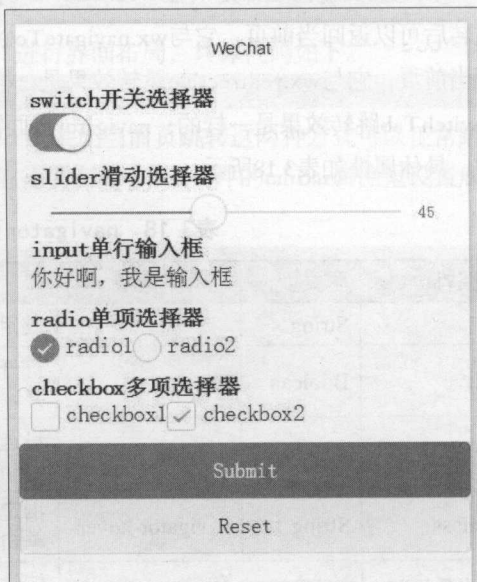


图3.29 填写表单

单击“Reset”按钮可以重置表单, 单击“Submit”按钮组件, 就可以把表单数据提交到js里进行处理, 提交数据如图3.30所示。

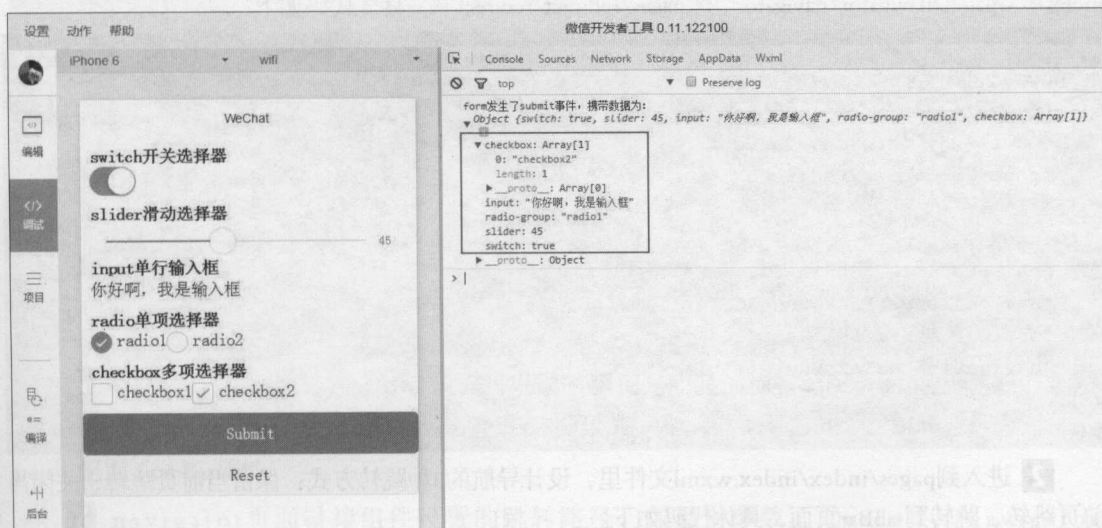


图3.30 表单提交数据

3.4 导航组件

微信小程序导航可以在页面中设置导航，可以使用navigator页面链接组件，也可以在js里设置导航进行页面跳转，同时可以设置导航条标题和显示动画效果。

3.4.1 navigator页面链接组件

navigator页面链接组件是用在WXML页面中跳转的导航，它有3种类型：第1种是保留当前页跳转，跳转后可以返回当前页，它与wx.navigateTo跳转效果是一样的；第2种是关闭当前页跳转，是无法返回当前页，它与wx.redirectTo跳转效果是一样的；第3种是跳转到底部标签导航指定的页面，它与wx.switchTab跳转效果是一样的；navigator页面链接组件的这些跳转效果都是通过open-type属性来控制的，具体属性如表3.18所示。

表3.18 navigator页面链接组件的属性

| 属性 | 类型 | 默认值 | 说明 |
|------------------|---------|-----------------|--|
| url | String | | 应用内的跳转链接 |
| redirect | Boolean | false | 打开方式为页面重定向，对应 wx.redirectTo（将被废弃，推荐使用 open-type） |
| open-type | String | navigate | 可选值“navigate”“redirect”“switchTab”，对应于 wx.navigateTo、wx.redirectTo、wx.switchTab的功能 |
| hover-class | String | navigator-hover | 指定单击时的样式类，当hover-class=“none”时，没有单击态效果 |
| hover-start-time | Number | 50 | 按住后多久出现单击态，单位为毫秒 |
| hover-stay-time | Number | 600 | 手指松开后单击态的保留时间，单位为毫秒 |

下面，我们来演示一下open-type不同导航类型的跳转效果。

1 新建一个navigator项目，进入到App.json文件，在pages属性里设置页面路径“pages/index/index”“pages/navigator/navigator”“pages/redirect/redirect”，具体代码如下。

```
{
  "pages": [
    "pages/index/index",
    "pages/navigator/navigator",
    "pages/redirect/redirect"
  ],
  "window": {
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#fff",
    "navigationBarTitleText": " 导航 ",
    "navigationBarTextStyle": "black"
  }
}
```

2 进入到pages/index/index.wxml文件里，设计导航的3种跳转方式：保留当前页跳转、关闭当前页跳转、跳转到tabBar页面，具体代码如下。

```
<view class="btn-area">
```



```
<navigator url="../../navigator/navigator?title=navigator" open-type="navigate" hover-
class="navigator-hover">wx.navigateTo 保留当前页跳转 </navigator>
<navigator url="../../redirect/redirect?title=redirect" open-type="redirect" hover-
class="other-navigator-hover">wx.redirectTo 关闭当前页跳转 </navigator>
<navigator url="../../redirect/redirect" open-type="switchTab" hover-class="other-
navigator-hover">wx.switchTab 跳转到 tabBar 页面 </navigator>
</view>
```

3 进入到pages/navigator/navigator.wxml文件里，进行界面布局，具体代码如下。

```
<view> 保留当前页进行跳转，单击左上角可以返回到当前页 </view>
```

4 进入到pages/redirect/redirect.wxml文件里，进行界面布局，具体代码如下。

```
<view> 关闭当前页进行跳转，跳转后无法返回到当前页 </view>
```

5 wx.navigateTo保留当前页跳转和wx.redirectTo关闭当前页跳转这两种方式可以正常跳转，但是wx.switchTab跳转到tabBar页面无法正常跳转，它需要在App.json文件的tabBar属性里设置底部标签导航，具体代码如下。

```
{
  "pages": [
    "pages/index/index",
    "pages/navigator/navigator",
    "pages/redirect/redirect"
  ],
  "window": {
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#fff",
    "navigationBarTitleText": " 导航 ",
    "navigationBarTextStyle": "black"
  },
  "tabBar": {
    "selectedColor": "red",
    "list": [
      {
        "pagePath": "pages/index/index",
        "text": " 首页 ",
        "iconPath": "iconPath",
        "selectedIconPath": "selectedIconPath"
      },
      {
        "pagePath": "pages/redirect/redirect",
        "text": " 当前页打开导航 ",
        "iconPath": "iconPath",
        "selectedIconPath": "selectedIconPath"
      }
    ]
  }
}
```

6 wx.switchTab跳转到tabBar页面可以跳转到指定的底部标签导航页面里，但是可以发现wx.navigateTo保留当前页跳转和wx.redirectTo关闭当前页跳转这两种方式无法跳转，是因为在app.json里配置了tabBar属性。

7 navigator页面链接组件设置的跳转路径，如果带参数，如url=“../../navigator/navigator?title=navigator”，title的值可以在跳转页面里js的onLoad函数里获得，具体代码如下。

```
Page({
  data:{},
  onLoad:function(options){
    console.log("title="+options);
  }
})
```

3.4.2 wx.navigateTo保留当前页跳转

wx.navigateTo保留当前页面，跳转到应用内的某个页面，使用wx.navigateBack可以返回到原页面，具体属性如表3.19所示。

表3.19 wx.navigateTo的属性

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--|
| url | String | 是 | 需要跳转的应用内非 tabBar 的页面的路径，路径后可以带参数。参数与路径之间使用“?”分隔，参数键与参数值用“=”相连，不同参数之间用“&”分隔，如“path?key=value&key2=value2” |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

1 进入pages/index/index.wxml文件，添加一个跳转按钮，保留当前页进行跳转，具体代码如下。

```
<view class="btn-area">
  <navigator url="../../navigator/navigator?title=navigator11" open-type="navigate" hover-class="navigator-hover">wx.navigateTo 保留当前页跳转 </navigator>
  <navigator url="../../redirect/redirect?title=redirect" open-type="redirect" hover-class="other-navigator-hover">wx.redirectTo 关闭当前页跳转 </navigator>
  <navigator url="../../redirect/redirect" open-type="switchTab" hover-class="other-navigator-hover">wx.switchTab 跳转到 tabBar 页面 </navigator>
  <button type="primary" bindtap="navigateBtn">保留当前页跳转 </button>
</view>
```

2 进入pages/index/index.js文件，添加一个navigateBtn事件函数，保留当前页跳转到pages/navigator/navigator.wxml页面里，具体代码如下。

```
Page({
  navigateBtn:function(){
    wx.navigateTo({
      url: '../../navigator/navigator',
      success: function(res){
        console.log(res);
      },
      fail: function() {
        // fail
      },
      complete: function() {
        // complete
      }
    })
  }
})
```

```
    })  
  }  
})
```

3.4.3 wx.redirectTo关闭当前页跳转

wx.redirectTo关闭当前页面，跳转到应用内的某个页面，具体属性如表3.20所示。

表3.20 wx.redirectTo的属性

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--|
| url | String | 是 | 需要跳转的应用内非 tabBar 的页面的路径，路径后可以带参数。参数与路径之间使用“?”分隔，参数键与参数值用“=”相连，不同参数之间用“&”分隔，如“path?key=value&key2=value2” |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

1 进入pages/index/index.wxml文件，添加一个跳转按钮，关闭当前页进行跳转，具体代码如下。

```
<view class="btn-area">  
  <navigator url="../../navigator/navigator?title=navigator11" open-type="navigate" hover-class="navigator-hover">wx.navigateTo 保留当前页跳转 </navigator>  
  <navigator url="../../redirect/redirect?title=redirect" open-type="redirect" hover-class="other-navigator-hover">wx.redirectTo 关闭当前页跳转 </navigator>  
  <navigator url="../../redirect/redirect" open-type="switchTab" hover-class="other-navigator-hover">wx.switchTab 跳转到 tabBar 页面 </navigator>  
  <button type="primary" bindtap="navigateBtn">保留当前页跳转 </button>  
  <button type="primary" bindtap="redirectBtn">关闭当前页跳转 </button>  
</view>
```

2 进入pages/index/index.js文件，添加一个redirectBtn事件函数，保留当前页跳转到pages/navigator/navigator.wxml页面里，具体代码如下。

```
Page({  
  navigateBtn:function(){  
    wx.navigateTo({  
      url: '../../navigator/navigator',  
      success: function(res){  
        console.log(res);  
      },  
      fail: function() {  
        // fail  
      },  
      complete: function() {  
        // complete  
      }  
    })  
  },  
  redirectBtn:function(){  
    wx.redirectTo({
```



```
url: '../navigator/navigator',
success: function(res){
    console.log(res);
},
fail: function() {
    // fail
},
complete: function() {
    // complete
}
})
}
```

3.4.4 wx.switchTab跳转到tabBar页面

跳转到 tabBar 页面，并关闭其他所有非 tabBar 页面，具体属性如表3.21所示。

表3.21 wx.switchTab的属性

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|---|
| url | String | 是 | 需要跳转的 tabBar 页面的路径（需在 App.json 的 tabBar 字段定义的页面），路径后不能带参数 |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

1 进入pages/index/index.wxml文件，添加一个跳转按钮，跳转到tabBar页面，具体代码如下。

```
<view class="btn-area">
  <navigator url="../../navigator/navigator?title=navigator11" open-type="navigate" hover-
class="navigator-hover">wx.navigateTo 保留当前页跳转 </navigator>
  <navigator url="../../redirect/redirect?title=redirect" open-type="redirect" hover-
class="other-navigator-hover">wx.redirectTo 关闭当前页跳转 </navigator>
  <navigator url="../../redirect/redirect" open-type="switchTab" hover-class="other-
navigator-hover">wx.switchTab 跳转到 tabBar 页面 </navigator>
  <button type="primary" bindtap="navigateBtn"> 保留当前页跳转 </button>
  <button type="primary" bindtap="redirectBtn"> 关闭当前页跳转 </button>
  <button type="primary" bindtap="switchBtn"> 跳转到 tabBar 页面 </button>
</view>
```

2 进入pages/index/index.js文件，添加一个navigateBtn事件函数，保留当前页跳转到pages/redirect/redirect.wxml页面里，具体代码如下。

```
Page({
  navigateBtn:function(){
    wx.navigateTo({
      url: '../navigator/navigator',
      success: function(res){
        console.log(res);
      },
      fail: function() {
```

```
        // fail
      },
      complete: function() {
        // complete
      }
    ))
  },
  redirectBtn:function(){
    wx.redirectTo({
      url: '../navigator/navigator',
      success: function(res){
        console.log(res);
      },
      fail: function() {
        // fail
      },
      complete: function() {
        // complete
      }
    ))
  },
  switchBtn:function(){
    wx.switchTab({
      url: '../redirect/redirect',
      success: function(res){
        // success
      },
      fail: function() {
        // fail
      },
      complete: function() {
        // complete
      }
    ))
  }
})
```

`wx.navigateTo` 和 `wx.redirectTo` 不允许跳转到 `tabbar` 页面，只能用 `wx.switchTab` 跳转到 `tabBar` 页面。

3.4.5 wx.navigateBack返回上一页

`wx.navigateBack` 关闭当前页面，返回上一页面或多级页面。可通过 `getCurrentPages()` 获取当前的页面栈，决定需要返回几层。具体属性如表3.22所示。

表3.22 `wx.navigateBack`的属性

| 属性 | 类型 | 必填 | 说明 |
|-------|--------|----|--------------------------------|
| delta | Number | 1 | 返回的页面数，如果 delta 大于现有页面数，则返回到首页 |

1 进入pages/navigator/navigator.wxml文件，添加一个返回按钮，单击返回按钮，可以返回到上一级页面，具体代码如下。

```
<view> 保留当前页进行跳转，单击左上角可以返回到当前页 </view>
<button type="primary" bindtap="backBtn"> 返回上一页 </button>
```

2 进入pages/navigator/navigator.js文件，添加backBtn事件返回函数，具体代码如下。

```
Page({
  data:{},
  onLoad:function(options){
    console.log("title="+options);
  },
  backBtn:function(){
    wx.navigateBack({
      delta: 1
    })
  }
})
```

3 在pages/index/index.wxml文件里，单击“保留当前页跳转”按钮，可以进行页面跳转，在跳转的页面中单击“返回上一页”按钮，可以返回到上一级页面，如图3.31和图3.32所示。

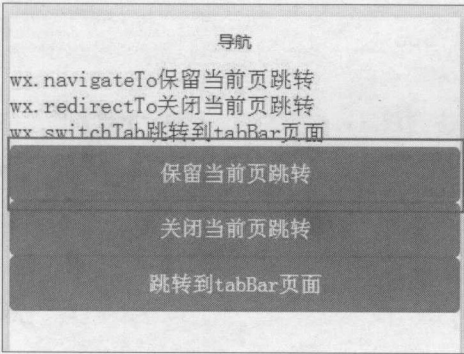


图3.31 index.wxml页面

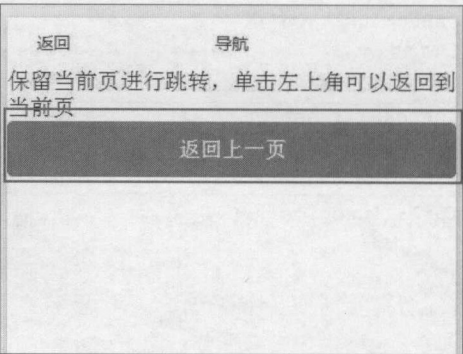


图3.32 navigator.wxml页面

3.4.6 设置导航条

wx.setNavigationBarTitle(OBJECT)动态设置当前页面的标题。具体属性如表3.23所示。

表3.23 wx.setNavigationBarTitle的属性

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| title | String | 是 | 页面标题 |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

1 进入pages/navigator/navigator.js文件，对页面窗口标题进行重新设计，具体代码如下。


```
Page({
  data:{},
  onLoad:function(options){
    console.log("title="+options);
    wx.setNavigationBarTitle({
      title: '新页面'
    })
  },
  backBtn:function(){
    wx.navigateBack({
      delta: 1
    })
  }
})
```

2 在pages/index/index.wxml文件里，单击“保留当前页跳转”按钮，可以进行页面跳转，跳转后标题变为新页面，如图3.33和图3.34所示。

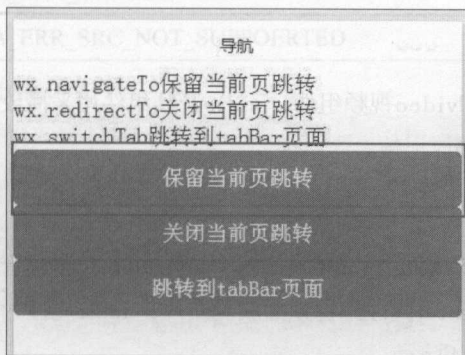


图3.33 index.wxml页面

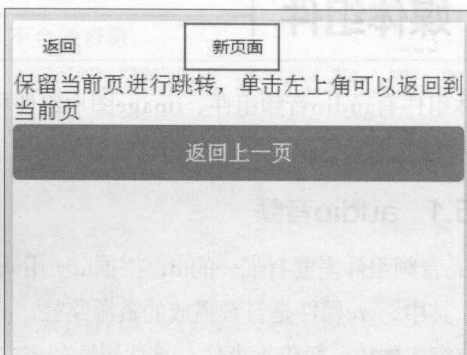


图3.34 navigator.wxml页面

3 在pages/navigator/navigator.js文件里，如果要在当前页面显示导航条加载动画，可以使用wx.showNavigationBarLoading()函数，具体代码如下。

```
Page({
  data:{},
  onLoad:function(options){
    console.log("title="+options);
    wx.setNavigationBarTitle({
      title: '新页面'
    });
    wx.showNavigationBarLoading();
  },
  backBtn:function(){
    wx.navigateBack({
      delta: 1
    })
  }
})
```

界面效果如图3.35所示。

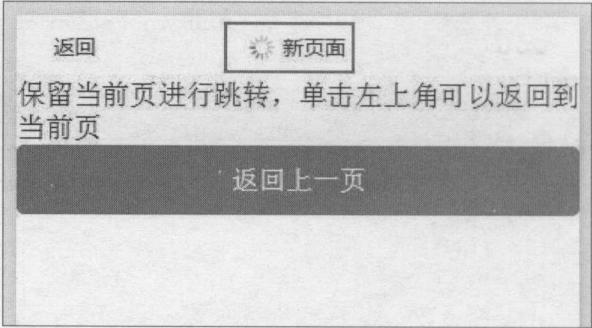


图3.35 导航加载效果

4 如果要当前页面显示导航条加载动画，可以使用wx.showNavigationBarLoading()函数；如果想隐藏导航条加载动画，可以使用wx.hideNavigationBarLoading()函数。

3.5 媒体组件

媒体组件有audio音频组件、image图片组件和video视频组件，audio音频组件用来播放音乐，image图片组件用来显示图片，video视频组件用来播放视频。

3.5.1 audio音频

audio音频组件需要有唯一的id，根据id使用wx.createAudioContext('myAudio')创建音频播放的环境，其中，src属性是音频播放的资源路径，poster属性是音频播放的图片，name属性为音频名称，还有绑定播放、暂停等事件，具体属性如表3.24所示。

表3.24 audio音频的属性

| 属性 | 类型 | 默认值 | 说明 |
|-----------|-------------|-------|--|
| id | String | | video 组件的唯一标识符 |
| src | String | | 要播放音频的资源地址 |
| loop | Boolean | false | 是否循环播放 |
| controls | Boolean | true | 是否显示默认控件 |
| poster | String | | 默认控件上的音频封面的图片资源地址，如果 controls 属性值为 false 则设置 poster 无效 |
| name | String | 未知音频 | 默认控件上的音频名字，如果 controls 属性值为 false 则设置 name 无效 |
| author | String | 未知作者 | 默认控件上的作者名字，如果 controls 属性值为 false 则设置 author 无效 |
| binderror | EventHandle | | 当发生错误时触发 error 事件，detail = {errMsg: MediaError.code} |
| bindplay | EventHandle | | 当开始/继续播放时触发play事件 |

续表

| 属性 | 类型 | 默认值 | 说明 |
|----------------|-------------|-----|--|
| bindpause | EventHandle | | 当暂停播放时触发 pause 事件 |
| bindtimeupdate | EventHandle | | 当播放进度改变时触发 timeupdate 事件, detail = {currentTime, duration} |
| bindended | EventHandle | | 当播放到末尾时触发 ended 事件 |

MediaError.code错误码如表3.25所示。

表3.25 错误码

| 错误码 | 说明 |
|------------------------------|-----------|
| MEDIA_ERR_ABORTED | 获取资源被用户禁止 |
| MEDIA_ERR_NETWORK | 网络错误 |
| MEDIA_ERR_DECODE | 解码错误 |
| MEDIA_ERR_SRC_NOT_SUPPOERTED | 不合适资源 |

示例代码如下。

```
<!-- audio.wxml -->
<audio poster="{{poster}}" name="{{name}}" author="{{author}}" src="{{src}}" id=
"myAudio" controls loop></audio>

<button type="primary" bindtap="audioPlay">播放</button>
<button type="primary" bindtap="audioPause">暂停</button>
<button type="primary" bindtap="audio14">设置当前播放时间为 14 秒</button>
<button type="primary" bindtap="audioStart">回到开头</button>
```

代码说明如图3.36所示。

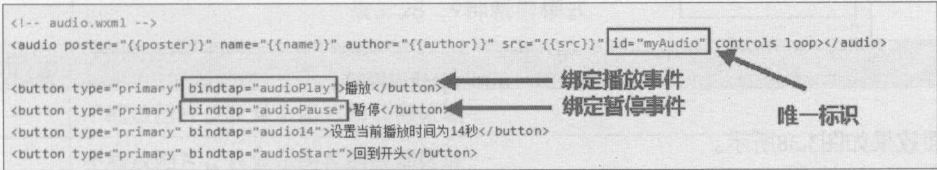


图3.36 audio.wxml代码说明

```
// audio.js
Page({
  onReady: function (e) {
    // 使用 wx.createAudioContext 获取 audio 上下文 context
    this.audioCtx = wx.createAudioContext('myAudio')
  },
  data: {
    poster: 'http://y.gtimg.cn/music/photo_new/T002R300x300M000003rsKF44GyaSk.jpg?max_age=2592000',
    name: '此时此刻',
    author: '许巍',
    src: 'http://ws.stream.qqmusic.qq.com/M500001VfvsJ21xFqb.mp3?guid=ffffffff82def4af
```



```
4b12b3cd9337d5e7&uin=346897220&vkey=6292F51E1E384E06DCBDC9AB7C49FD713D632D313AC4858BAC
B8DDD29067D3C601481D36E62053BF8DFEAF74C0A5CCFADD6471160CAF3E6A&fromtag=46',
},
audioPlay: function () {
  this.audioCtx.play()
},
audioPause: function () {
  this.audioCtx.pause()
},
audio14: function () {
  this.audioCtx.seek(14)
},
audioStart: function () {
  this.audioCtx.seek(0)
}
})
```

代码说明如图3.37所示。

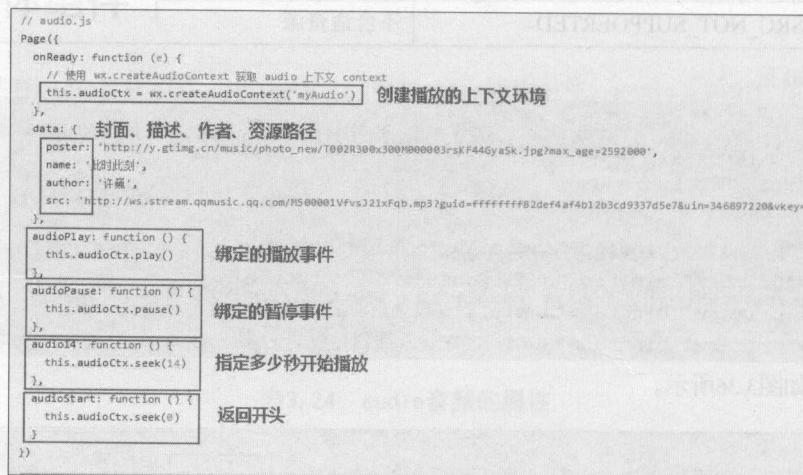


图3.37 audio.js代码说明

界面效果如图3.38所示。

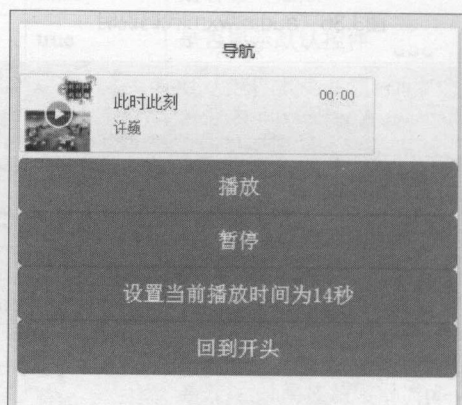


图3.38 音频播放界面效果

3.5.2 image图片

image图片组件有两类展现模式：一类是缩放模式，在缩放模式里又包括4种方式；另一类是裁剪模式，在裁剪模式里又包括9种方式，具体属性如表3.26所示。

表3.26 image图片的属性

| 属性 | 类型 | 默认值 | 说明 |
|-----------|-------------|---------------|--|
| src | String | | 图片资源地址 |
| mode | String | 'scaleToFill' | 图片裁剪、缩放的模式 |
| binderror | HandleEvent | | 当错误发生时，发布到 AppService 的事件名，事件对象 event.detail = {errMsg: 'something wrong'} |
| bindload | HandleEvent | | 当图片载入完毕时，发布到 AppService 的事件名，事件对象 event.detail = {height:'图片高度px', width:'图片宽度px'} |

通过mode属性来设置4种缩放模式，如表3.27所示。

表3.27 4种缩放模式

| 模式 | 说明 |
|-------------|--|
| scaleToFill | 不保持纵横比缩放图片，使图片的宽高完全拉伸至填满 image 元素 |
| aspectFit | 保持纵横比缩放图片，使图片的长边能完全显示出来。也就是说，可以完整地将图片显示出来 |
| aspectFill | 保持纵横比缩放图片，只保证图片的短边能完全显示出来。也就是说，图片通常只在水平或垂直方向是完整的，另一个方向将会发生截取 |
| widthFix | 宽度不变，高度自动变化，保持原图宽高比不变 |

通过mode属性来设置9种裁剪模式，如表3.28所示。

表3.28 9种裁剪模式

| 模式 | 说明 |
|--------------|-------------------|
| top | 不缩放图片，只显示图片的顶部区域 |
| bottom | 不缩放图片，只显示图片的底部区域 |
| center | 不缩放图片，只显示图片的中间区域 |
| left | 不缩放图片，只显示图片的左边区域 |
| right | 不缩放图片，只显示图片的右边区域 |
| top left | 不缩放图片，只显示图片的左上边区域 |
| top right | 不缩放图片，只显示图片的右上边区域 |
| bottom left | 不缩放图片，只显示图片的左下边区域 |
| bottom right | 不缩放图片，只显示图片的右下边区域 |

示例代码如下。

```
<view class="page">
  <view class="page__hd">
    <text class="page__title">image</text>
    <text class="page__desc"> 图片 </text>
  </view>
  <view class="page__bd">
    <view class="section section_gap" wx:for="{{array}}" wx:for-item="item">
      <view class="section__title">{{item.text}}</view>
      <view class="section__ctn">
        <image style="width: 200px; height: 200px; background-color: #eeeeee;" mode=
          "{{item.mode}}" src="{{src}}"></image>
      </view>
    </view>
  </view>
</view>
</view>
```

```
Page({
  data: {
    array: [{
      mode: 'scaleToFill',
      text: 'scaleToFill: 不保持纵横比缩放图片，使图片完全适应'
    }, {
      mode: 'aspectFit',
      text: 'aspectFit: 保持纵横比缩放图片，使图片的长边能完全显示出来'
    }, {
      mode: 'aspectFill',
      text: 'aspectFill: 保持纵横比缩放图片，只保证图片的短边能完全显示出来'
    }, {
      mode: 'top',
      text: 'top: 不缩放图片，只显示图片的顶部区域'
    }, {
      mode: 'bottom',
      text: 'bottom: 不缩放图片，只显示图片的底部区域'
    }, {
      mode: 'center',
      text: 'center: 不缩放图片，只显示图片的中间区域'
    }, {
      mode: 'left',
      text: 'left: 不缩放图片，只显示图片的左边区域'
    }, {
      mode: 'right',
      text: 'right: 不缩放图片，只显示图片的右边区域'
    }, {
      mode: 'top left',
      text: 'top left: 不缩放图片，只显示图片的左上边区域'
    }, {
      mode: 'top right',
      text: 'top right: 不缩放图片，只显示图片的右上边区域'
    }, {
      mode: 'bottom left',
```



```
text: 'bottom left: 不缩放图片, 只显示图片的左下边区域',
}, {
  mode: 'bottom right',
  text: 'bottom right: 不缩放图片, 只显示图片的右下边区域',
}],
src: '../../resources/cat.jpg'
},
imageError: function(e) {
  console.log('image3 发生 error 事件, 携带值为 ', e.detail.errMsg)
}
})
```

图片效果如图3.39~图3.51所示。



图3.39 原图

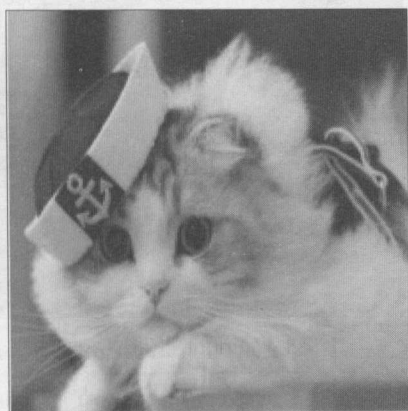


图3.40 scaleToFill缩放模式

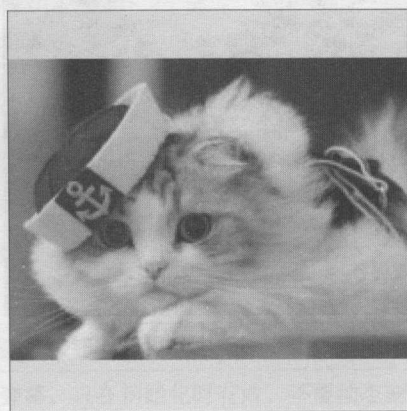


图3.41 aspectFit缩放模式

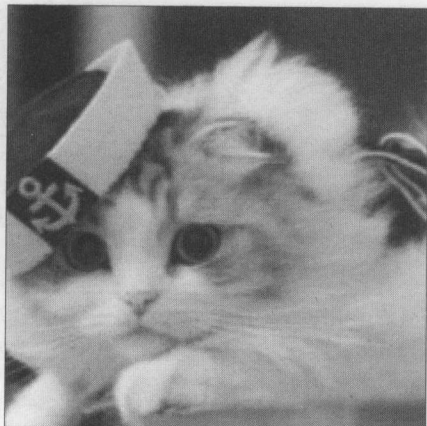


图3.42 aspectFill缩放模式

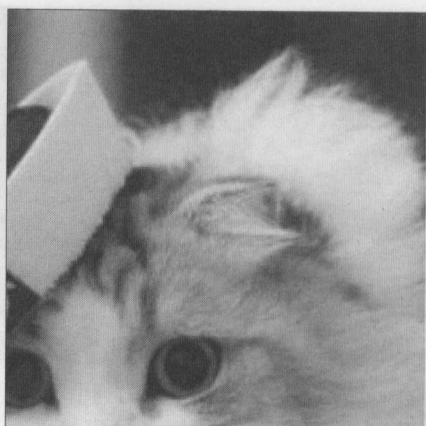


图3.43 top裁剪模式

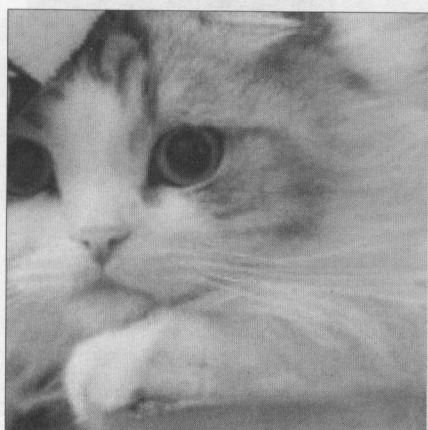


图3.44 bottom裁剪模式

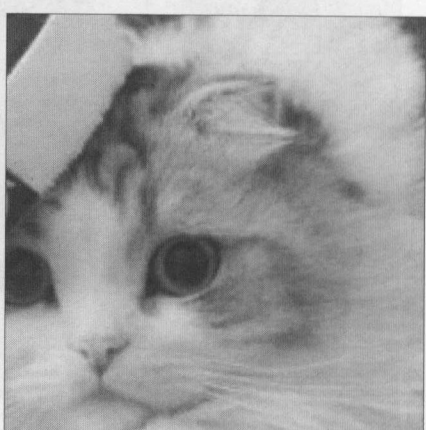


图3.45 center裁剪模式

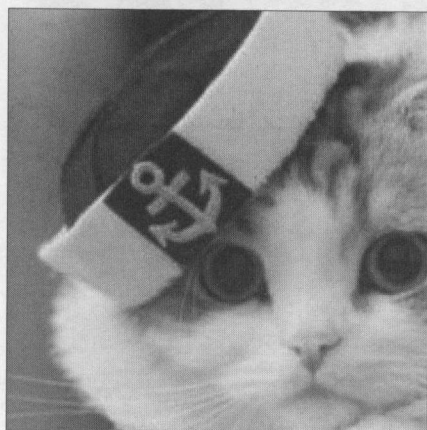


图3.46 left裁剪模式

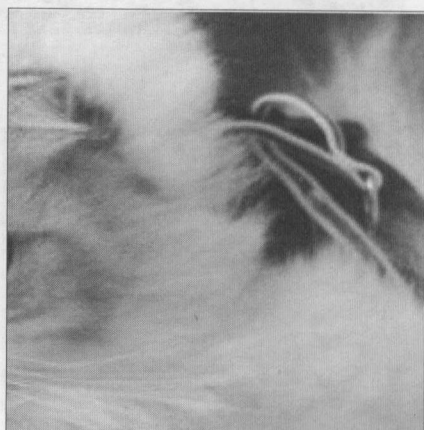


图3.47 right裁剪模式

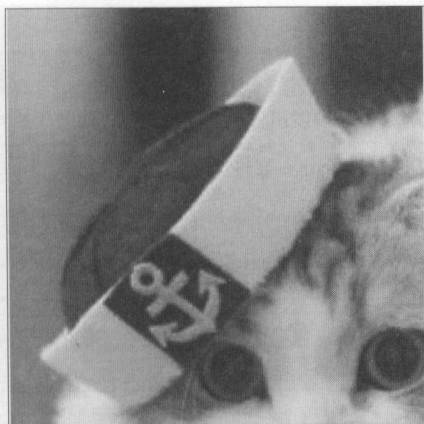


图3.48 top left裁剪模式

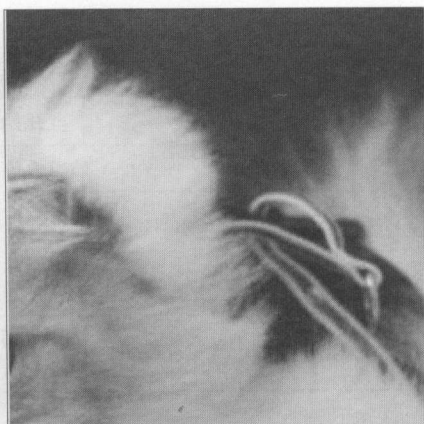


图3.49 top right裁剪模式



图3.50 bottom left裁剪模式



图3.51 bottom right裁剪模式

3.5.3 video视频

video视频组件是用来播放视频的组件，这个组件可以控制是否显示默认播放控件（播放/暂停按钮、播放进度、时间），可以发送弹幕信息等功能，video组件的默认宽度为300px、高度为225px，设置宽高需要通过wxss设置width和height，具体属性如表3.29所示。

表3.29 video视频的属性

| 属性 | 类型 | 默认值 | 说明 |
|--------------|---------|-------|-----------------------------|
| src | String | | 要播放视频的资源地址 |
| controls | Boolean | true | 是否显示默认播放控件（播放/暂停按钮、播放进度、时间） |
| danmu-list | Object | Array | 弹幕列表 |
| danmu-btn | Boolean | false | 是否显示弹幕按钮，只在初始化时有效，不能动态变更 |
| enable-danmu | Boolean | false | 是否展示弹幕，只在初始化时有效，不能动态变更 |
| autoplay | Boolean | false | 是否自动播放 |

续表

| 属性 | 类型 | 默认值 | 说明 |
|----------------|-------------|---------|---|
| bindplay | EventHandle | | 当开始/继续播放时触发play事件 |
| bindpause | EventHandle | | 当暂停播放时触发 pause 事件 |
| bindended | EventHandle | | 当播放到末尾时触发 ended 事件 |
| bindtimeupdate | EventHandle | | 播放进度变化时触发，event.detail = {currentTime: ‘当前播放时间’ }。触发频率应该在 250ms/次 |
| objectFit | String | contain | 当视频大小与 video 容器大小不一致时视频的表现形式。 contain: 包含，fill: 填充，cover: 覆盖 |

示例代码如下。

```
<view class="section tc">
  <video id="myVideo" src="http://wxsnsdy.tc.qq.com/105/20210/snsdyvideodownload?filekey=30280201010421301f0201690402534804102ca905ce620b1241b726bc41dcff44e00204012882540400&bizid=1023&hy=SH&fileparam=302c020101042530230204136ffd93020457e3c4ff02024ef202031e8d7f02030f42400204045a320a0201000400" danmu-list="{{danmuList}}" enable-danmu danmu-btn controls></video>
  <view class="btn-area">
    <button bindtap="bindButtonTap"> 获取视频 </button>
    <input bindblur="bindInputBlur"/>
    <button bindtap="bindSendDanmu"> 发送弹幕 </button>
  </view>
</view>
```

```
function getRandomColor () {
  let rgb = []
  for (let i = 0 ; i < 3; ++i){
    let color = Math.floor(Math.random() * 256).toString(16)
    color = color.length == 1 ? '0' + color : color
    rgb.push(color)
  }
  return '#' + rgb.join('')
}

Page({
  onReady: function (res) {
    this.videoContext = wx.createVideoContext('myVideo')
  },
  inputValue: '',
  data: {
    src: '',
    danmuList: [
      {
        text: '第 1s 出现的弹幕',
        color: '#ff0000',
        time: 1
      },
    ],
  },
})
```

```
{
  text: '第 3s 出现的弹幕',
  color: '#ff00ff',
  time: 3
}]
},
bindInputBlur: function(e) {
  this.inputValue = e.detail.value
},
bindButtonTap: function() {
  var that = this
  wx.chooseVideo({
    sourceType: ['album', 'camera'],
    maxDuration: 60,
    camera: ['front', 'back'],
    success: function(res) {
      that.setData({
        src: res.tempFilePath
      })
    }
  })
},
bindSendDanmu: function () {
  this.videoContext.sendDanmu({
    text: this.inputValue,
    color: getRandomColor()
  })
}
})
})
```

界面效果如图3.52所示。

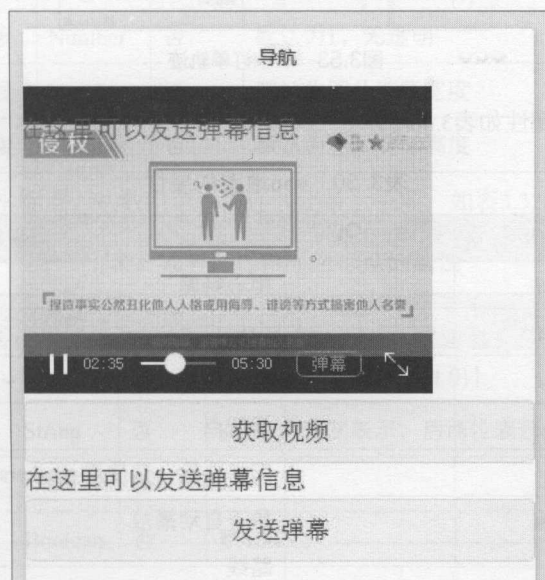


图3.52 视频播放界面

3.6 地图组件

map地图组件用来开发与地图有关的应用，如地图导航、打车软件、京东商城的订单轨迹都会用到地图组件。在地图上可以标记覆盖物以及指定一系列的坐标位置，如京东的仓库和客户的收货地址，如图3.53所示。



图3.53 京东订单轨迹

map地图组件的具体属性如表3.30所示。

表3.30 map地图的属性

| 属性 | 类型 | 默认值 | 说明 |
|-----------|---------|-----|------------------|
| longitude | Number | | 中心经度 |
| latitude | Number | | 中心纬度 |
| scale | Number | 16 | 缩放级别，取值范围为5~18 |
| markers | Array | | 标记点 |
| covers | Array | | 即将移除，请使用 markers |
| autoplay | Boolean | | 是否自动播放 |
| polyline | Array | | 路线 |

续表

| 属性 | 类型 | 默认值 | 说明 |
|------------------|-------------|-----|-----------------|
| circles | Array | | 圆 |
| controls | Array | | 控件 |
| include-points | Array | | 缩放视野以包含所有给定的坐标点 |
| show-location | Boolean | | 显示带有方向的当前定位点 |
| bindmarkertap | EventHandle | | 单击标记点时触发 |
| bindcontroltap | EventHandle | | 单击控件时触发 |
| bindregionchange | EventHandle | | 视野发生变化时触发 |
| bindtap | EventHandle | | 单击地图时触发 |

markers用于在地图上显示标记的位置，如表3.31所示。

表3.31 markers地图标记的属性

| 属性 | 说明 | 类型 | 必填 | 备注 |
|-----------|--------|--------|----|---------------------------------------|
| id | 标记点id | Number | 否 | marker单击事件回调会返回此id |
| latitude | 纬度 | Number | 是 | 浮点数，范围为-90 ~ 90 |
| longitude | 经度 | Number | 是 | 浮点数，范围为-180 ~ 180 |
| title | 标注点名 | String | 否 | |
| iconPath | 显示的图标 | String | 是 | 项目目录下的图片路径，支持相对路径写法，以“/”开头则表示相对小程序根目录 |
| rotate | 旋转角度 | Number | 否 | 顺时针旋转的角度，范围为0 ~ 360，默认为 0 |
| alpha | 标注的透明度 | Number | 否 | 默认为1，无透明 |
| width | 标注图标宽度 | Number | 否 | 默认为图片实际宽度 |
| height | 标注图标高度 | Number | 否 | 默认为图片实际高度 |

polyline指定一系列坐标点，从数组第一项连线至最后一项，如表3.32所示。

表3.32 polyline坐标点的属性

| 属性 | 说明 | 类型 | 必填 | 备注 |
|------------|-------|---------|----|----------------------------------|
| points | 经纬度数组 | Array | 是 | [{latitude: 0, longitude: 0}] |
| color | 线的颜色 | String | 否 | 8位十六进制表示，后两位表示alpha值，如：#000000AA |
| width | 线的宽度 | Number | 否 | |
| dottedLine | 是否虚线 | Boolean | 否 | 默认false |

circles在地图上显示圆，如表3.33所示。

表3.33 circles显示圆的属性

| 属性 | 说明 | 类型 | 必填 | 备注 |
|-------------|-------|--------|----|----------------------------------|
| latitude | 纬度 | Number | 是 | 浮点数，范围为-90~90 |
| longitude | 经度 | Number | 是 | 浮点数，范围为-180~180 |
| color | 描边的颜色 | String | 否 | 8位十六进制表示，后两位表示alpha值，如：#000000AA |
| fillColor | 填充颜色 | String | 否 | 8位十六进制表示，后两位表示alpha值，如：#000000AA |
| radius | 半径 | Number | 是 | |
| strokeWidth | 描边的宽度 | Number | 否 | |

controls在地图上显示控件，控件不随地图移动，如表3.34所示。

表3.34 controls显示控件的属性

| 属性 | 说明 | 类型 | 必填 | 备注 |
|-----------|----------|---------|----|---------------------------------------|
| id | 控件id | Number | 否 | 控件单击事件回调会返回此id |
| position | 控件在地图的位置 | Object | 是 | 控件相对地图位置 |
| iconPath | 显示的图标 | String | 是 | 项目目录下的图片路径，支持相对路径写法，以“/”开头则表示相对小程序根目录 |
| clickable | 是否可单击 | Boolean | 否 | 默认不可单击 |

position控件的位置是相对地图的位置，如表3.35所示。

表3.35 position控件位置的属性

| 属性 | 说明 | 类型 | 必填 | 备注 |
|--------|------------|--------|----|---------|
| left | 距离地图的左边界多远 | Number | 否 | 默认为0 |
| top | 距离地图的上边界多远 | Number | 否 | 默认为0 |
| width | 控件宽度 | Number | 否 | 默认为图片宽度 |
| height | 控件高度 | Number | 否 | 默认为图片高度 |



注意：地图组件的经纬度必填，如果不填经纬度则默认值是北京的经纬度。

示例代码如下。

```
<!-- map.wxml -->
<map id="map" longitude="113.324520" latitude="23.099994" scale="14" controls=
"{{controls}}" bindcontrolltap="controlltap" markers="{{markers}}" bindmarkertap=
"markertap" polyline="{{polyline}}" bindregionchange="regionchange" show-location
```

```
style="width: 100%; height: 300px;"></map>
```

```
// map.js
Page({
  data: {
    markers: [{
      iconPath: "/resources/others.png",
      id: 0,
      latitude: 23.099994,
      longitude: 113.324520,
      width: 50,
      height: 50
    }],
    polyline: [{
      points: [{
        longitude: 113.3245211,
        latitude: 23.10229
      }, {
        longitude: 113.324520,
        latitude: 23.21229
      }],
      color: "#FF0000DD",
      width: 2,
      dottedLine: true
    }],
    controls: [{
      id: 1,
      iconPath: '/resources/location.png',
      position: {
        left: 0,
        top: 300 - 50,
        width: 50,
        height: 50
      },
      clickable: true
    }]
  },
  regionchange(e) {
    console.log(e.type)
  },
  markertap(e) {
    console.log(e.markerId)
  },
  controltap(e) {
    console.log(e.controlId)
  }
})
```

界面效果如图3.54所示。

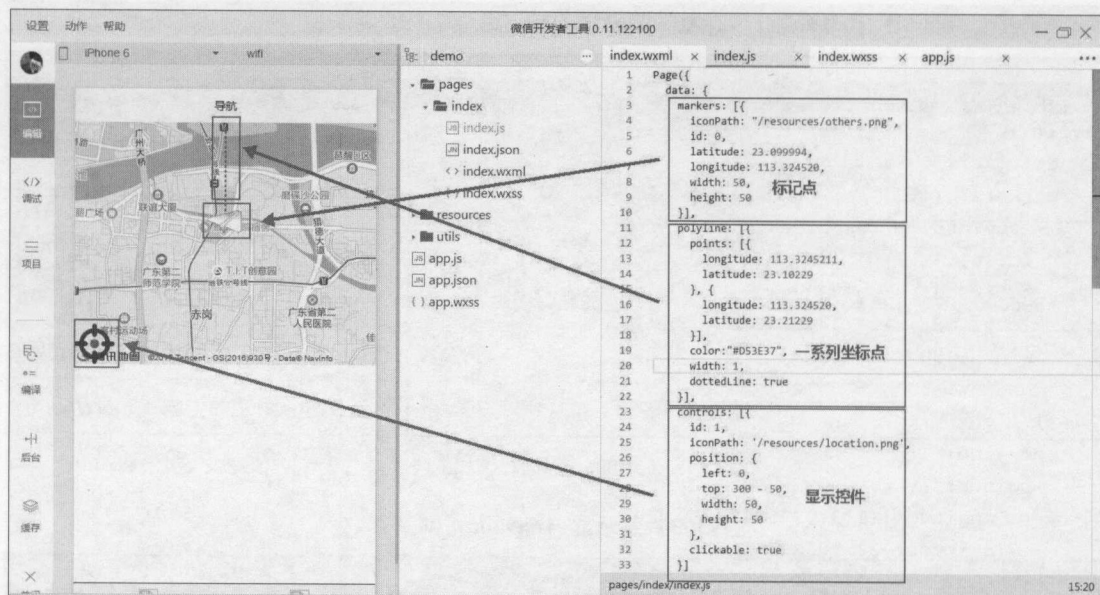


图3.54 地图界面效果

3.7 画布组件

canvas画布组件用来绘制正方形、圆形或者其他的形状，如图3.55所示。

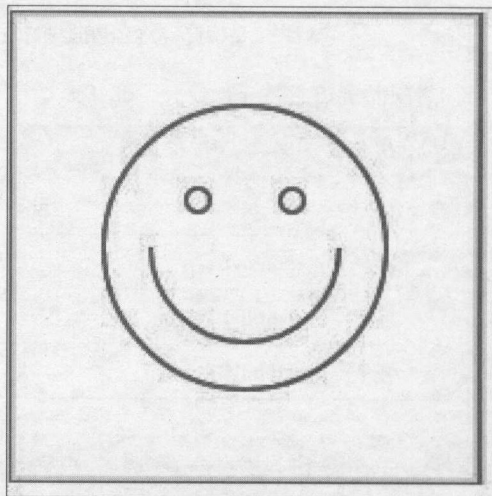


图3.55 canvas画布组件绘制图形

canvas画布组件的默认宽度为300px、高度为225px，在使用的时候需要有唯一的标识。它有手指触摸动作开始、手指触摸后移动、手指触摸动作结束、手指触摸动作被打断等事件，具体属性如表3.36所示。

表3.36 canvas画布的属性

| 属性 | 类型 | 默认值 | 说明 |
|-----------------|-------------|-------|--|
| canvas-id | String | | canvas 组件的唯一标识符 |
| disable-scroll | Boolean | false | 当在 canvas 中移动时, 禁止屏幕滚动以及下拉刷新 |
| bindtouchstart | EventHandle | | 手指触摸动作开始 |
| bindtouchmove | EventHandle | | 手指触摸后移动 |
| bindtouchend | EventHandle | | 手指触摸动作结束 |
| bindtouchcancel | EventHandle | | 手指触摸动作被打断, 如来电提醒、弹窗 |
| bindlongtap | EventHandle | | 手指长按 500ms 之后触发, 触发了长按事件后进行移动不会触发屏幕的滚动 |
| binderror | EventHandle | | 当发生错误时触发 error 事件, detail = {errMsg: 'something wrong' } |

示例代码如下。

```
<!-- canvas.wxml -->
<canvas style="width: 300px; height: 200px;" canvas-id="firstCanvas"></canvas>
<!-- 当使用绝对定位时, 文档流后边的 canvas 的显示层级高于前边的 canvas -->
<canvas style="width: 400px; height: 500px;" canvas-id="secondCanvas"></canvas>
<!-- 因为 canvas-id 与前一个 canvas 重复, 该 canvas 不会显示, 并会发送一个错误事件到 AppService -->
<canvas style="width: 400px; height: 500px;" canvas-id="secondCanvas" binderror="canvasIdErrorCallback"></canvas>
// canvas.js
Page({
  canvasIdErrorCallback: function (e) {
    console.error(e.detail.errMsg)
  },
  onReady: function (e) {

    // 使用 wx.createContext 获取绘图上下文 context
    var context = wx.createContext()

    context.setStrokeStyle("#00ff00")
    context.setLineWidth(5)
    context.rect(0, 0, 200, 200)
    context.stroke()
    context.setStrokeStyle("#ff0000")
    context.setLineWidth(2)
    context.moveTo(160, 100)
    context.arc(100, 100, 60, 0, 2 * Math.PI, true)
    context.moveTo(140, 100)
    context.arc(100, 100, 40, 0, Math.PI, false)
    context.moveTo(85, 80)
    context.arc(80, 80, 5, 0, 2 * Math.PI, true)
    context.moveTo(125, 80)
    context.arc(120, 80, 5, 0, 2 * Math.PI, true)
```

```

context.stroke()

// 调用 wx.drawCanvas，通过 canvasId 指定在哪张画布上绘制，通过 actions 指定绘制行为
wx.drawCanvas({
  canvasId: 'firstCanvas',
  actions: context.getActions() // 获取绘图动作数组
})
}
})

```

代码说明如图3.56所示。

| | |
|--|--------------------------|
| <pre>onReady: function (e) {</pre> | |
| <pre>// 使用 wx.createContext 获取绘图上下文 context</pre> | |
| <pre>var context = wx.createContext()</pre> | 创建画布绘图环境 |
| <pre>context.setStrokeStyle("#00ff00") context.setLineWidth(5) context.rect(0, 0, 200, 200) context.stroke()</pre> | 设置矩形位置、大小、边框颜色、线宽 |
| <pre>context.setStrokeStyle("#ff0000") context.setLineWidth(2)</pre> | 设置线条颜色和线宽 |
| <pre>context.moveTo(160, 100) context.arc(100, 100, 60, 0, 2 * Math.PI, true)</pre> | 移动画笔坐标位置，绘制弧形 |
| <pre>context.moveTo(140, 100) context.arc(100, 100, 40, 0, Math.PI, false)</pre> | 移动画笔坐标位置，绘制弧形 |
| <pre>context.moveTo(85, 80) context.arc(80, 80, 5, 0, 2 * Math.PI, true)</pre> | 移动画笔坐标位置，绘制弧形 |
| <pre>context.moveTo(125, 80) context.arc(120, 80, 5, 0, 2 * Math.PI, true) context.stroke()</pre> | 移动画笔坐标位置，绘制弧形 |
| <pre>// 调用 wx.drawCanvas，通过 canvasId 指定在哪张画布上绘制，通过 actions 指定绘制行为</pre> | |
| <pre>wx.drawCanvas({</pre> | |
| <pre> canvasId: 'firstCanvas',</pre> | 画布唯一标识 |
| <pre> actions: context.getActions()</pre> | 绘图动作集合 |
| <pre>})</pre> | |
| <pre>}</pre> | |

图3.56 绘图代码说明

画布组件以及画布相关API会在后续的章节中深入地学习。

3.8 沙场大练兵：表单登录注册微信小程序

微信小程序里有丰富的表单组件，可通过这些组件的使用，来完成京东登录界面、手机号注册界面、企业注册界面的注册微信小程序的设计，如图3.57~图3.59所示。

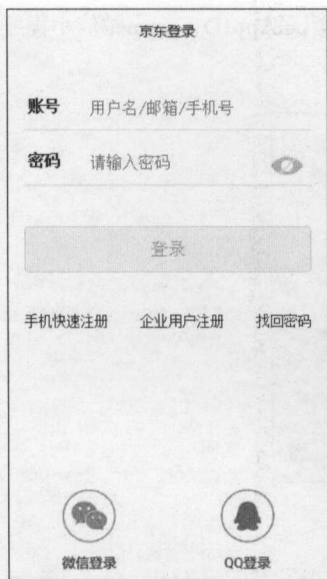


图3.57 登录

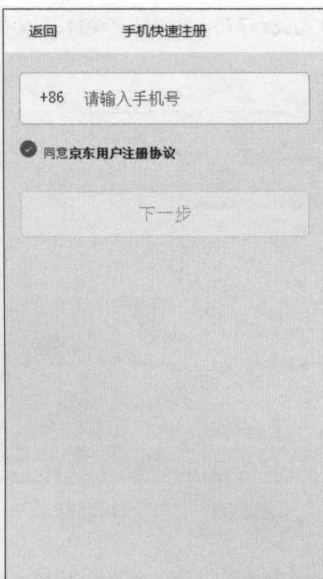


图3.58 手机快速注册

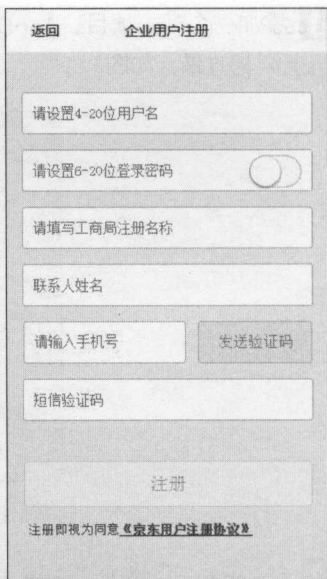


图3.59 企业用户注册

会用到view视图容器组件、button按钮组件、image图片组件、input输入框组件、checkbox多项选择器组件、switch开关选择器、navigator页面链接组件等的使用，将这些组件进行界面的布局设计即可完成表单登录注册设计。

3.8.1 登录设计

在登录表单里，输入账号、密码进行登录，在账号、密码输入框里都有友好的提示信息；“登录”按钮默认是灰色不可用状态，只有在输入内容后，才变为可用状态；在按钮的下面提供手机快速注册、企业用户注册、找回密码链接；最下面是微信、QQ第三方登录方式，如图3.60所示。

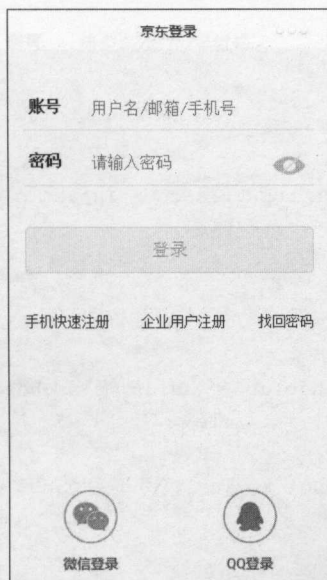


图3.60 登录界面

1 添加一个form项目，AppID为wxa7730e0596be9404，只有填写AppID，form微信小程序才能在手机上浏览效果，如图3.61所示。

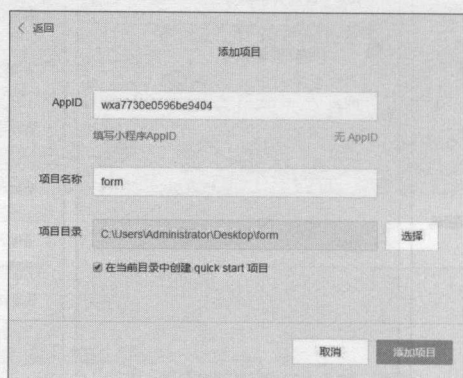


图3.61 添加form项目

2 在App.json里添加“pages/login/login”“pages/mobile/mobile”“pages/company/company”3个文件目录，并删除默认的文件目录以及相应的文件夹，如图3.62所示。

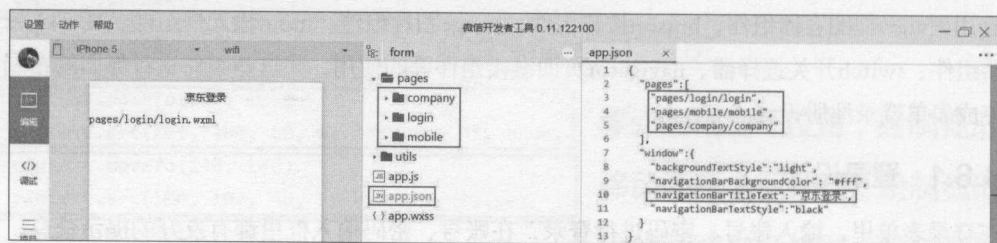


图3.62 App.json配置

3 在“pages/login/login”文件里，进行账号、密码输入框的布局设计，并添加相应的样式，代码如下。

```
login.wxml
<view class="content">
  <view class="account">
    <view class="title">账号</view>
    <view class="num"><input bindinput="accountInput" placeholder="用户名 / 邮箱 / 手机号 "
placeholder-style="color: #999999;"/></view>
  </view>
  <view class="hr"></view>
  <view class="account">
    <view class="title">密码</view>
    <view class="num"><input bindblur="pwdBlur" placeholder=" 请输入密码 " password
placeholder-style="color: #999999;"/></view>
    <view class="see">
      <image src="/images/see.jpg" style="width:42px;height:30px;"/></image>
    </view>
  </view>
  <view class="hr"></view>
</view>
```

```
login.wxss
.content{
  margin-top: 40px;
}
.account{
  display: flex;
  flex-direction: row;
  padding-left: 20px;
  padding-top: 20px;
  padding-bottom: 10px;
  width: 90%;
}
.title{
  margin-right: 30px;
  font-weight: bold;
}
.hr{
  border: 1px solid #cccccc;
  opacity: 0.2;
  width: 90%;
  margin: 0 auto;
}
.see{
  position: absolute;
  right: 20px;
}
```

界面效果如图3.63所示。

京东登录

账号 用户名/邮箱/手机号

密码 请输入密码

图3.63 输入框布局设计

4 在“pages/login/login”文件里，进行“登录”按钮、手机快速注册、企业用户注册、密码以及第三方登录的布局，并添加相应的样式，代码如下。

```
login.wxml
<view class="content">
  <view class="account">
    <view class="title">账号</view>
    <view class="num"><input bindinput="accountInput" placeholder="用户名 / 邮箱 / 手机号"
placeholder-style="color:#999999;"/></view>
  </view>
  <view class="hr"></view>
  <view class="account">
    <view class="title">密码</view>
    <view class="num"><input bindblur="pwdBlur" placeholder="请输入密码" password
placeholder-style="color:#999999;"/></view>
    <view class="see">
      <image src="/images/see.jpg" style="width:42px;height:30px;"/></image>
    </view>
  </view>
  <view class="hr"></view>
  <button class="btn" disabled="{{disabled}}" type="{{btnstate}}" bindtap="login">登录</button>
  <view class="operate">
    <view><navigator url="../mobile/mobile">手机快速注册</navigator></view>
    <view><navigator url="../company/company">企业用户注册</navigator></view>
    <view>找回密码</view>
  </view>
  <view class="login">
    <view><image src="/images/wxlogin.png" style="width:70px;height:98px;"/></image></view>
    <view><image src="/images/qqlogin.png" style="width:70px;height:98px;"/></image></view>
  </view>
</view>
```

```
login.wxss
.content{
  margin-top: 40px;
}
.account{
  display: flex;
  flex-direction: row;
  padding-left: 20px;
  padding-top: 20px;
  padding-bottom: 10px;
  width: 90%;
}
.title{
```

```
margin-right: 30px;
font-weight: bold;
}
.hr{
border: 1px solid #cccccc;
opacity: 0.2;
width: 90%;
margin: 0 auto;
}
.see{
position: absolute;
right: 20px;
}
.btn{
width: 90%;
margin-top: 40px;
color: #999999;
}
.operate{
display: flex;
flex-direction: row;
}
.operate view{
margin: 0 auto;
margin-top: 40px;
font-size: 14px;
color: #333333;
}
.login{
display: flex;
flex-direction: row;
margin-top: 150px;
}
.login view{
margin: 0 auto;
}
```

界面效果如图3.64所示。

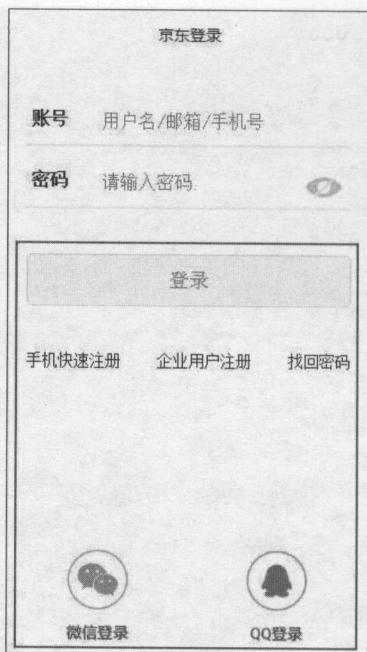


图3.64 布局设计

5 在“pages/login/login”文件里，在js中添加accountInput、pwdBlur事件函数，当“账号”输入框里输入内容后，“登录”按钮变为可用状态，代码如下。

```
login.js
Page({
  data:{
    disabled:true,
    btnstate:"default",
    account:"",
    password:""
  },
  accountInput:function(e){
    var content = e.detail.value;
    console.log(content);
    if(content != ''){
      this.setData({disabled:false,btnstate:"primary",account:content});
    }else{
      this.setData({disabled:true,btnstate:"default"});
    }
  },
  pwdBlur:function(e){
    var password = e.detail.value;
    if(password != ''){
      this.setData({password:password});
    }
  }
})
```

界面效果如图3.65所示。

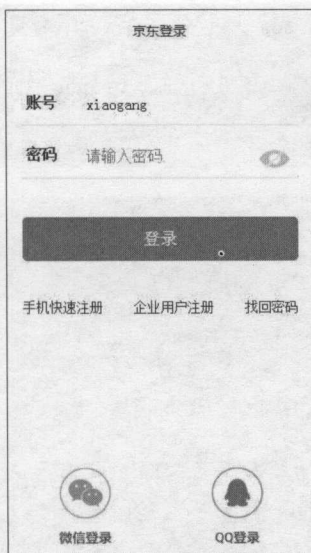


图3.65 按钮可用状态

3.8.2 手机号注册设计

在手机号注册里，需要设计输入框用来输入手机号，设计同意注册协议及“下一步”按钮，如图3.66所示。

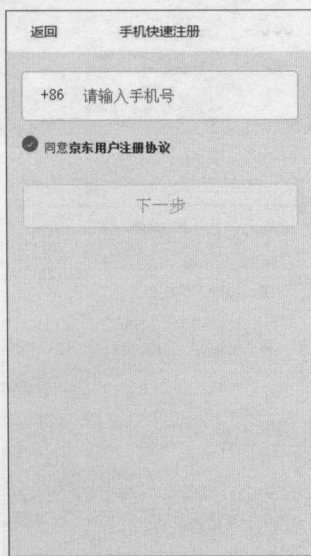


图3.66 手机号注册界面

1 在“pages/mobile/mobile”文件里，进行手机号输入框的布局设计，并添加相应的样式，代码如下。

```
mobile.wxml
<view class="content">
  <view class="hr"></view>
```

```
<view class="numbg">
  <view>+86</view>
  <view><input placeholder=" 请输入手机号 " maxlength="11" bindblur="mobileblur"/></view>
</view>
```

```
mobile.wxss
.content{
  width:100%;
  height: 600px;
  background-color: #f2f2f2;
}
.hr{
  padding-top:20px;
}
.numbg{
  border: 1px solid #cccccc;
  width: 90%;
  margin: 0 auto;
  background-color: #ffffff;
  border-radius: 5px;
  display: flex;
  flex-direction: row;
  height: 50px;
}
.numbg view{
  margin-left: 20px;
  margin-top:14px;
}
```

界面效果如图3.67所示。

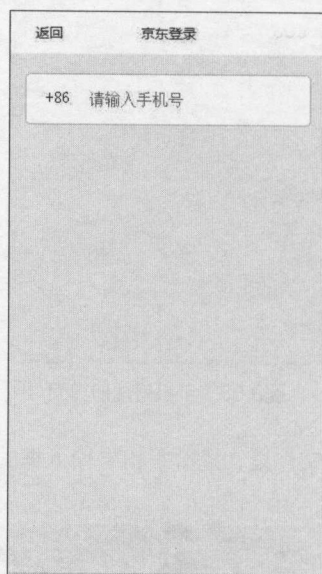


图3.67 手机号输入框

2 在“pages/mobile/mobile”文件里，设计注册协议和“下一步”按钮，并添加相应的样式，代码如下。

```
mobile.wxml
<view class="content">
  <view class="hr"></view>
  <view class="numbg">
    <view>+86</view>
    <view><input placeholder=" 请输入手机号 " maxlength="11" bindblur="mobileblur"/></view>
  </view>
  <view>
    <view class="xieyi">
      <icon type="success" color="red" size="18"></icon>
      <text class="agree"> 同意 </text>
      <text class="opinion"> 京东用户注册协议 </text>
    </view>
  </view>
  <button class="btn" disabled="{{disabled}}" type="{{btnstate}}" bindtap="login">下一步</button>
</view>
```

```
mobile.wxss
.content{
  width:100%;
  height: 600px;
  background-color: #f2f2f2;
}
.hr{
  padding-top:20px;
}
.numbg{
  border: 1px solid #cccccc;
  width: 90%;
  margin: 0 auto;
  background-color: #ffffff;
  border-radius: 5px;
  display: flex;
  flex-direction: row;
  height: 50px;
}
.numbg view{
  margin-left: 20px;
  margin-top:14px;
}
.xieyi{
  margin-top:15px;
  margin-left:15px;
}
.agree{
  font-size: 13px;
```



```
margin-left: 5px;
color: #666666;
}
.opinion{
font-size: 13px;
color: #000000;
font-weight: bold;
}
.btn{
width:90%;
margin-top:30px;
}
```

界面效果如图3.68所示。

图3.68 注册协议及“下一步”按钮

3 在“pages/mobile/mobile”文件里，添加mobileblur事件，如果输入手机号，“下一步”按钮则变为可用状态，代码如下。

```
mobile.js
Page({
  data:{
    disabled:true,
```

```
    btnstate:"default",  
    mobile:""  
  },  
  mobileblur:function(e){  
    var content = e.detail.value;  
    if(content != "")(  
      this.setData({disabled:false,btnstate:"primary",mobile:content});  
    )else(  
      this.setData({disabled:true,btnstate:"defalut",mobile:""});  
    )  
  }  
})
```

界面效果如图3.69所示。

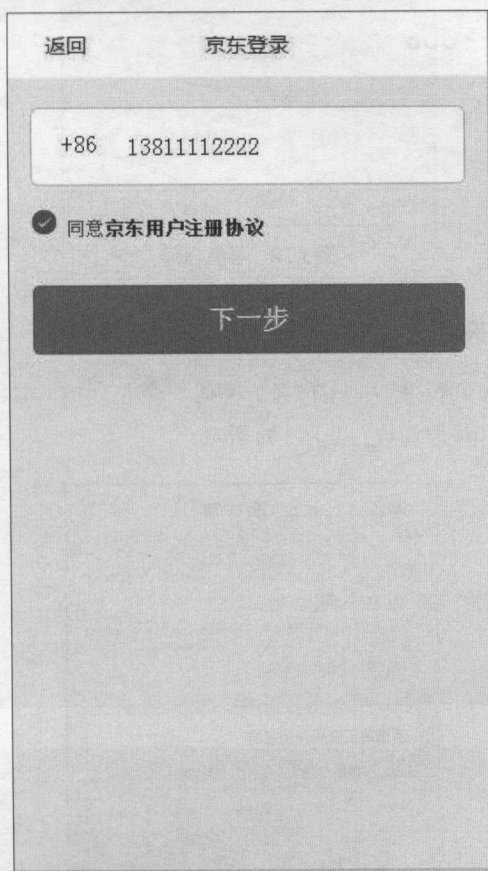


图3.69 “下一步”按钮可用

4 在“mobile.json”文件里，添加“navigationBarTitleText”“手机快速注册”属性，设置导航标题为“手机快速注册”，如图3.70所示。

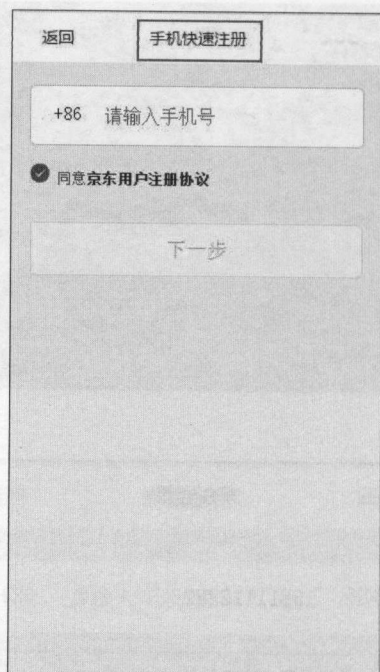


图3.70 导航标题

3.8.3 企业用户注册设计

在企业用户注册里，有6个表单项：用户名、密码、企业全称、联系人姓名、手机号、短信验证码，有一个“注册”按钮和注册协议，如图3.71所示。

图3.71 企业用户注册界面

1 在“pages/company/company”文件里，进行用户名、密码、企业全称、联系人姓名、手机号、短信验证码表单项的布局设计，并添加相应的样式，代码如下。

```
company.wxml
<form bindsubmit="formSubmit" bindreset="formReset">
<view class="content">
  <view class="hr"></view>
  <view class="item">
    <input type="text" name="loginName" placeholder="请设置 4-20 位用户名" placeholder-
class="holder" bindblur="accountblur"/>
  </view>
  <view class="item flex">
    <input type="text" password name="password" placeholder="请设置 6-20 位登录密码"
placeholder-class="holder"/>
    <switch type="switch" name="switch"/>
  </view>
  <view class="item">
    <input type="text" name="company" placeholder="请填写工商局注册名称" placeholder-
class="holder" />
  </view>
  <view class="item">
    <input type="text" name="userName" placeholder="联系人姓名" placeholder-class=
"holder" />
  </view>
  <view class="mobileInfo">
    <view class="mobile">
      <input type="text" name="mobile" placeholder="请输入手机号" placeholder-class=
"holder" />
    </view>
    <view class="code">发送验证码</view>
  </view>
  <view class="item">
    <input type="text" name="code" placeholder="短信验证码" placeholder-class="holder" />
  </view>
</view>
</form>
```

```
company.wxss
.content{
  width: 100%;
  height: 700px;
  background-color: #f2f2f2;
}
.hr{
  padding-top:40px;
}
.item{
  margin: 0 auto;
  border: 1px solid #cccccc;
  height: 40px;
```

```

        width: 90%;
        border-radius: 3px;
        background-color: #ffffff;
        margin-bottom: 15px;
    }
    .item input{
        height: 40px;
        line-height: 40px;
        margin-left: 10px;
    }
    .holder{
        font-size: 14px;
        color: #999999;
    }
    .flex{
        display: flex;
        flex-direction: row;
    }
    .flex input{
        width: 300px;
    }
    .item switch{
        margin-top: 5px;
        margin-right: 5px;
    }
    .mobileInfo{
        display: flex;
        flex-direction: row;
    }
    .mobile{
        margin: 0 auto;
        border: 1px solid #cccccc;
        height: 40px;
        width: 50%;
        border-radius: 3px;
        background-color: #ffffff;
        margin-bottom: 15px;
        display: flex;
        flex-direction: row;
        margin-left: 5%;
    }
    .mobile input{
        margin-top: 8px;
        margin-left: 10px;
    }
    .code{
        border: 1px solid #cccccc;
        height: 40px;
        width: 35%;
    }

```

```

background-color: #EFEFEE;
border-radius: 3px;
text-align: center;
margin-left: 10px;
line-height: 40px;
color: #999999;
font-size: 15px;
margin-bottom: 15px;
margin-right: 5%;
}

```

界面效果如图3.72所示。

返回 京东登录

请设置4-20位用户名

请设置6-20位登录密码

请填写工商局注册名称

联系人姓名

请输入手机号 发送验证码

短信验证码

图3.72 企业用户注册表单项

2 在“pages/company/company”文件里，设计“注册”按钮和同意注册协议，并添加相应的样式，代码如下。

```

company.wxml
<form bindsubmit="formSubmit" bindreset="formReset">
<view class="content">
  <view class="hr"></view>
  <view class="item">
    <input type="text" name="loginName" placeholder="请设置4-20位用户名" placeholder-
class="holder" bindblur="accountblur"/>
  </view>

```



```

<view class="item flex">
  <input type="text" password name="password" placeholder=" 请设置 6-20 位登录密码 "
placeholder-class="holder"/>
  <switch type="switch" name="switch"/>
</view>
<view class="item">
  <input type="text" name="company" placeholder=" 请填写工商局注册名称 " placeholder-
class="holder" />
</view>
<view class="item">
  <input type="text" name="userName" placeholder=" 联系人姓名 " placeholder-class=
"holder" />
</view>
<view class="mobileInfo">
  <view class="mobile">
    <input type="text" name="mobile" placeholder=" 请输入手机号 " placeholder-class=
"holder" />
  </view>
  <view class="code"> 发送验证码 </view>
</view>
<view class="item">
  <input type="text" name="code" placeholder=" 短信验证码 " placeholder-class="holder" />
</view>
<button class="btn" disabled="{{disabled}}" type="{{btnstate}}" form-type="submit">
注册</button>
<view class="xieyi">
  <text class="agree"> 注册即视为同意 </text><text class="opinion">《京东用户注册协议》
</text>
</view>
</view>
</form>

```

```

.content{
  width: 100%;
  height: 700px;
  background-color: #f2f2f2;
}
.hr{
  padding-top:40px;
}
.item{
  margin: 0 auto;
  border: 1px solid #cccccc;
  height: 40px;
  width: 90%;
  border-radius: 3px;
  background-color: #ffffff;
  margin-bottom:15px;
}
.item input{

```

```
height: 40px;
line-height: 40px;
margin-left: 10px;
}
.holder{
  font-size: 14px;
  color: #999999;
}
.flex{
  display: flex;
  flex-direction: row;
}

.flex input{
  width: 300px;
}
.item switch{
  margin-top: 5px;
  margin-right: 5px;
}
.mobileInfo{
  display: flex;
  flex-direction: row;
}
.mobile{
  margin: 0 auto;
  border: 1px solid #cccccc;
  height: 40px;
  width: 50%;
  border-radius: 3px;
  background-color: #ffffff;
  margin-bottom: 15px;
  display: flex;
  flex-direction: row;
  margin-left: 5%;
}
.mobile input{
  margin-top: 8px;
  margin-left: 10px;
}
.code{
  border: 1px solid #cccccc;
  height: 40px;
  width: 35%;
  background-color: #EFEFEE;
  border-radius: 3px;
  text-align: center;
  margin-left: 10px;
  line-height: 40px;
  color: #999999;
}
```

```
font-size: 15px;
margin-bottom: 15px;
margin-right: 5%;
}
.btn{
width: 90%;
color: #999999;
margin-top: 40px;
}
.xieyi{
margin-top: 15px;
margin-left: 15px;
font-size: 13px;
}
}
.agree{
margin-left: 5px;
color: #666666;
}
}
.opinion{
color: red;
font-weight: bold;
text-decoration: underline;
}
}
```

界面效果如图3.73所示。

返回

京东登录

请设置4-20位用户名

请设置6-20位登录密码

请填写工商局注册名称

联系人姓名

请输入手机号

发送验证码

短信验证码

注册

注册即视为同意[《京东用户注册协议》](#)

图3.73 “注册”按钮

3 当输入用户名后，“注册”按钮变为可用状态，同时将表单内容提交到company.js后台，保存到缓存中，代码如下。

```
Page({
  data: {
    disabled: true,
    btnstate: "default"
  },
  accountblur: function(e) {
    var content = e.detail.value;
    if (content != "") {
      this.setData({ disabled: false, btnstate: "primary" });
    } else {
      this.setData({ disabled: true, btnstate: "default" });
    }
  },
  formSubmit: function(e) {
    console.log(e);
    var user = new Object();
    user.account = e.detail.value.loginName;
    user.password = e.detail.value.password;
    user.company = e.detail.value.company;
    user.userName = e.detail.value.userName;
    user.code = e.detail.value.code;
    user.mobile = e.detail.value.mobile;
    user.switch = e.detail.value.switch;
    wx.setStorageSync('user', user);
    wx.showToast({
      title: "注册成功",
      icon: "success",
      duration: 1000,
      success: function() {
        wx.navigateTo({
          url: '../login/login'
        })
      }
    });
  }
})
```

4 在“company.json”文件里，添加“navigationBarTitleText”：“企业用户注册”属性，设置导航标题为“企业用户注册”。

5 单击“项目”导航，然后单击“预览”按钮，可以在手机上预览微信小程序，扫描二维码，也可以上传微信小程序，如图3.74所示。

| | | | |
|----------|--------|---|---|
| header | String | 否 | 设置小程序的header，header中不能设置Referer |
| method | String | 否 | 请求方法，支持：OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT |
| dataType | String | 否 | 设置请求数据的类型，如果设置了dataType为json，则会尝试对响应的数据进行JSON.parse |

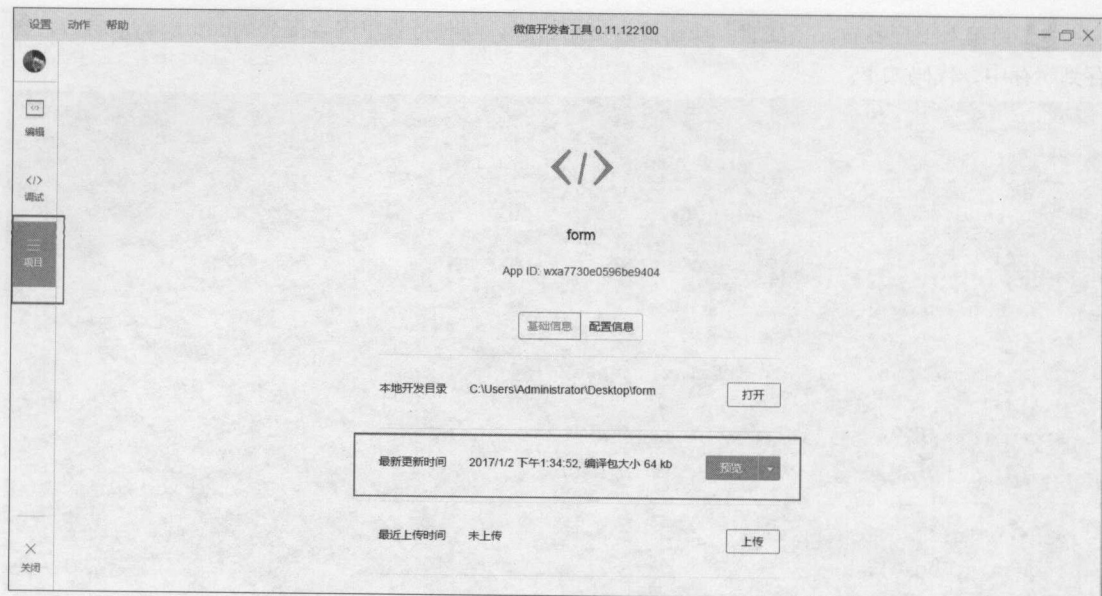


图3.74 预览微信小程序

这样就完成了京东登录注册表单微信小程序的设计，在登录界面进行登录，可以通过手机快速注册来进行手机号注册，也可以通过企业用户注册来注册企业账号。

3.9 小结

本章主要学习了微信小程序组件的使用，重点掌握以下内容。

- 1** 掌握视图容器组件的使用，会制作海报轮播效果、页签切换效果、上下滑动效果以及左右滑动效果。
- 2** 掌握基础内容组件的使用，包括图标组件、文本组件以及进度条组件的使用。
- 3** 掌握表单组件的使用，利用表单组件来设计微信小程序的表单内容，可以提交表单以及重置表单内容。
- 4** 掌握导航组件的使用，保留当前页跳转以及关闭当前页跳转。
- 5** 掌握媒体组件的使用，包括音频组件、图片组件以及视频组件的使用。
- 6** 掌握地图组件和画布组件的使用。

第4章 必备的微信小程序API

微信小程序提供了很多在开发微信小程序时会用到的API接口，有请求服务器数据、文件上传与下载、WebSocket会话、图片处理、文件操作、数据缓存、位置信息、设备应用、交互反馈、绘图、登录、微信支付、客服信息、分享等这些常用的API接口。

4.1 请求服务器数据API

`wx.request`是用来请求服务器数据的API，它发起的是HTTPS请求，同时它需要在微信公众平台配置HTTPS服务器域名，一个月内可申请3次修改，否则在有AppID创建的项目无法使用`wx.request`请求服务器数据的API，WebSocket会话、文件上传下载服务器域名都是如此，配置服务器域名如图4.1所示。

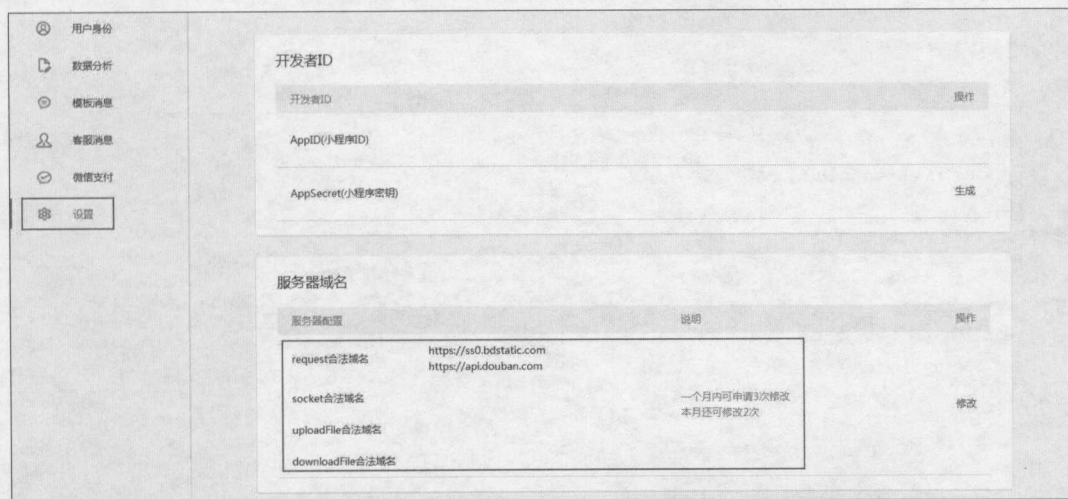


图4.1 配置服务器域名

`wx.request(object)`参数说明如表4.1所示。

表4.1 `wx.request`参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|---------------|----|--|
| url | String | 是 | 开发者服务器接口地址 |
| data | Object、String | 否 | 请求的参数 |
| header | Object | 否 | 设置请求的 header，header 中不能设置 Referer |
| method | String | 否 | 默认为 GET，有效值：OPTIONS、GET、HEAD、POST、PUT、DELETE、TRACE、CONNECT |
| dataType | String | 否 | 默认为 json。如果设置了 dataType 为 json，则会尝试对响应的数据做一次 JSON.parse |

续表

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|---|
| success | Function | 否 | 收到开发者服务器成功返回的回调函数，res = {data: ‘开发者服务器返回的内容’} |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

下面，我们来演示wx.request请求服务器数据API的使用。

1. 请求HTTP服务器数据

在js文件的onLoad函数里，使用wx.request请求猫眼电影HTTP服务器数据，具体代码如下。

```
Page({
  onLoad:function(){
    wx.request({
      url: 'http://m.maoyan.com/movie/list.json',
      data: {
        type:'hot',
        offset:0,
        limit:1000
      },
      method: 'GET',
      success: function(res){
        console.log(res);
      },
      fail: function() {
        // fail
      },
      complete: function() {
        // complete
      }
    })
  }
})
```

请求错误信息如图4.2所示。

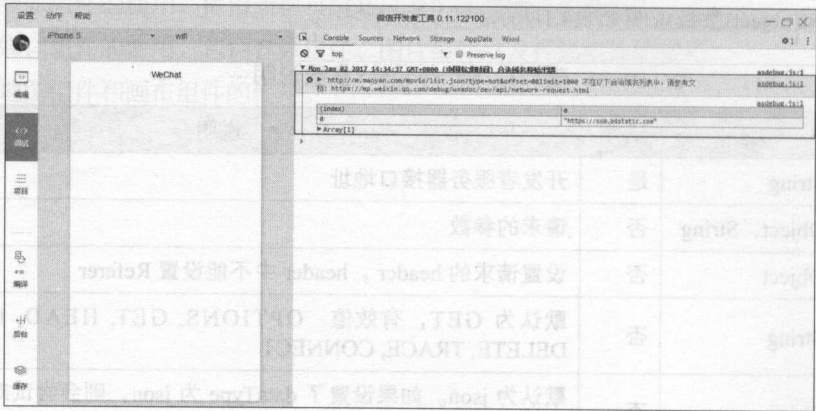


图4.2 HTTP请求

从图4.2中可以看出, `wx.request`无法请求HTTP域名的服务器, 访问服务器路径的时候, 会到公众开发平台里去找我们配置的HTTPS服务器域名, 如果域名存在, 就让访问, 否则不让访问。

`data` 数据说明, 最终发送给服务器的数据是 `String` 类型, 如果传入的 `data` 不是 `String` 类型, 会被转换成 `String`类型, 转换规则如下。

对于 `header['content-type']` 为 `'Application/json'` 的数据, 会对数据进行 `JSON` 序列化。

对于 `header['content-type']` 为 `'Application/x-www-form-urlencoded'` 的数据, 会将数据转换成 `query string` (`encodeURIComponent(k)=encodeURIComponent(v)&encodeURIComponent(k)=encodeURIComponent(v)...`)。

2. 请求HTTPS服务器数据

具体代码如下。

```
Page({
  onLoad:function(){
    wx.request({
      url: 'https://api.douban.com/v2/movie/in_theaters',
      data: {
      },
      method: 'GET',
      header: {
        'content-type': 'Application/json'
      },
      success: function(res){
        console.log(res);
      }
    })
  }
})
```

服务器返回数据如图4.3所示。

`content-type` 默认为 `'Application/json'`, 客户端的 `HTTPS TLS` 版本为1.2, 但 `Android` 的部分机型还未支持 `TLS 1.2`, 所以请确保 `HTTPS` 服务器的 `TLS` 版本支持1.2及以下版本; 要注意 `method` 的 `value` 必须为大写 (如`GET`) ; `url` 中不能有端口; `request` 的默认超时时间和最大超时时间都是 60s, `request` 的最大并发数是 5, 网络请求的 `referer` 是不可以设置的, 格式固定为 `https://servicewechat.com/{Appid}/{version}/page-frame.html`, 其中 `{Appid}` 为小程序的 `Appid`, `{version}` 为小程序的版本号, 版本号为 0 表示为开发版。



注意: 如果项目没有填写 `AppID`, 是可以访问 `HTTP` 请求以及公众开发平台以外的一些服务器请求的, 但是不填写 `AppID`, 在手机上是无法预览和使用的, 所以在学习过程中可以不填写 `AppID`, 来学习这些 `API` 的使用。

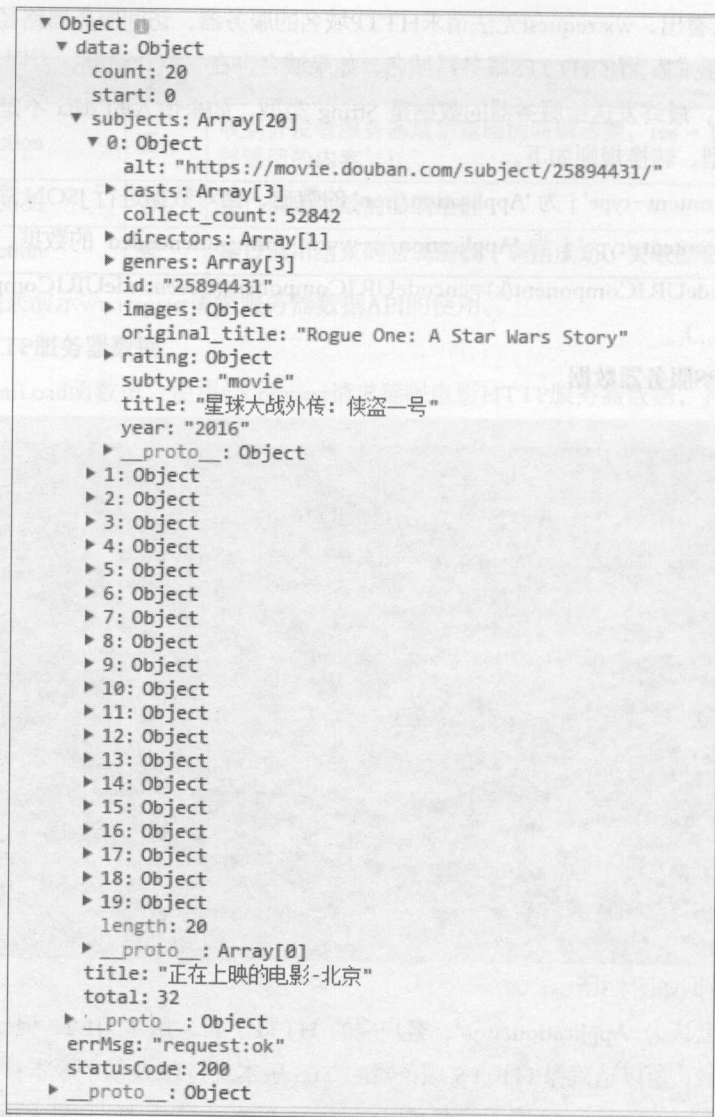


图4.3 服务器返回数据

4.2 文件上传与下载API

文件上传与下载API是我们经常会用到的API，它可以用来与服务器进行文件的上传与下载，如微信小程序客户端向服务器传输一些图片，或者从服务器那里获得图片，这时就可以使用文件上传与下载API，其请求服务器地址也需要在微信公众平台中进行配置。

微课视频



文件上传与下载API

4.2.1 wx.uploadFile文件上传

wx.uploadFile(object)参数说明如表4.2所示。

表4.2 wx.uploadFile参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|---|
| url | String | 是 | 开发者服务器 url |
| filePath | String | 是 | 要上传文件资源的路径 |
| name | String | 是 | 文件对应的 key，开发者在服务器端通过这个 key 可以获取到文件二进制内容 |
| header | Object | 否 | HTTP 请求 Header，header 中不能设置 Referer |
| formData | Object | 否 | HTTP 请求中其他额外的 form data |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

下面，我们来演示一下wx.uploadFile文件上传的使用，将选择的图片传到服务器里。

1 创建一个无AppID的项目，在微信小程序中选中上传的图片，利用wx.uploadFile上传图片到服务器，具体代码如下。

```
Page({
  onLoad: function() {
    wx.chooseImage({
      count: 9, // 最多可以选择的图片张数，默认为9
      sizeType: ['original', 'compressed'], // original 原图, compressed 压缩图，默认二者
      sourceType: ['album', 'camera'], // album 从相册选图, camera 使用相机，默认二者
      success: function(res) {
        var tempFilePaths = res.tempFilePaths;
        wx.uploadFile({
          url: 'http://localhost:8555/wxApp/WxUploadFileServlet',
          filePath: tempFilePaths[0],
          name: 'name',
          header: {
            'content-type': 'Application/json'
          },
          formData: {
            imgName: '我是图片名称',
            imgSize: '122kb'
          },
          success: function(res) {
            console.log(res);
          }
        })
      }
    })
  }
})
```

2 服务器端采用java代码来编写接收文件上传过来的图片信息，将图片保存到服务器中，具体代码如下。

```
package com.xiaogang.App.servlet;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.commons.fileupload.FileItem;
import org.apache.commons.fileupload.disk.DiskFileItemFactory;
import org.apache.commons.fileupload.servlet.ServletFileUpload;

/**
 *
 * @ClassName: WxUploadFileServlet
 * @Description: 用于接收微信小程序上传的图片
 * @author LG
 * @date 2016年12月5日 下午1:30:14
 */
public class WxUploadFileServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public WxUploadFileServlet() {
        super();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        doPost(request, response);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        try {
            DiskFileItemFactory factory = new DiskFileItemFactory();
            ServletFileUpload upload = new ServletFileUpload(factory);
            upload.setHeaderEncoding("UTF-8");
            List items = upload.parseRequest(request);
            Map param = new HashMap();
            for(Object object:items){
                FileItem fileItem = (FileItem) object;
                if (fileItem.isFormField()) {
```

```

        param.put(fileItem.getFieldName(), fileItem.getString
("utf-8"));
    }else{
        if("file".equals(fileItem.getFieldName())){
            String fileName = fileItem.getName();
            InputStream is = fileItem.getInputStream();
            // 创建 img 文件夹
            new File("E:/img/").mkdir();
            File file = new File("E:/img/",fileName);
            file.createNewFile();

            FileOutputStream fos = new FileOutputStream

(file);

            byte[] b = new byte[1024];
            while((is.read(b)) != -1){
                fos.write(b);
            }
            is.close();
            fos.close();
        }
    }
    System.out.println(param);
} catch (Exception e) {
    e.printStackTrace();
}
}
}

```

3 文件上传后，利用wx.uploadFile里的success回调函数，可以查看文件是否上传成功，如图4.4所示。

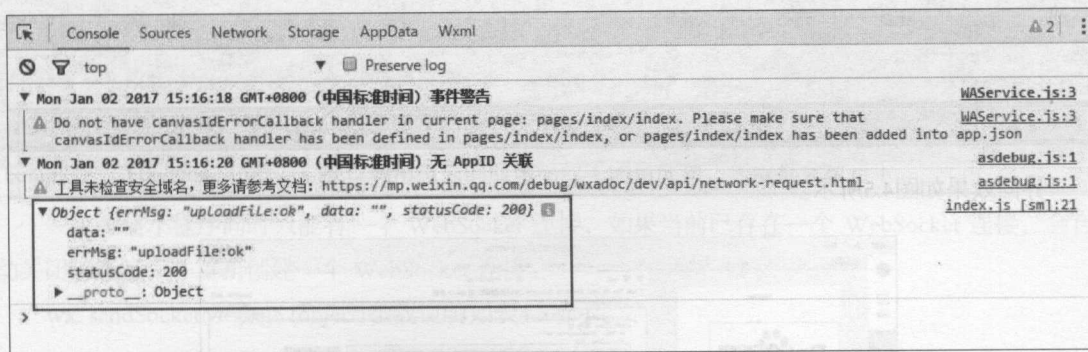


图4.4 回调函数返回值

4.2.2 wx.downloadFile文件下载

wx.uploadFile是文件上传的API，wx.downloadFile是文件下载的API，正好相反，它是从服务器获得数据，将数据下载到微信小程序客户端本地，参数说明如表4.3所示。

complete Function 否 返回下载文件的本地文件路径。调用成功，函数都会执行。

表4.3 wx.downloadFile参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|---|
| url | String | 是 | 开发者服务器 url |
| header | Object | 否 | HTTP 请求 Header，header 中不能设置 Referer |
| success | Function | 否 | 收到开发者服务器成功返回的回调函数，res = {data: ‘开发者服务器返回的内容’} |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

下面，我们来演示一下wx.downloadFile文件下载接口的使用。服务器传递一张图片给微信小程序客户端，将其下载到本地，显示出来。

1 在WXML文件里，添加image组件，用来显示服务器传递过来的图片，具体代码如下。

```
<image src="{{src}}" style="width:270px;height:126px;"></image>
```

2 在js文件里，下载服务器的一张图片，将它的临时路径赋值给src，具体代码如下。

```
Page({
  data:{
    src:''
  },
  onLoad:function(){
    var page = this;
    wx.downloadFile({
      url: "https://ss0.bdstatic.com/5aV1bjqh_Q23odCf/static/superman/img/logo/bd_logo1_31bdc765.png",
      type: 'image', // 下载资源的类型，用于客户端识别处理，有效值：image/audio/video
      success: function(res){
        console.log(res);
        var tempPath = res.tempFilePath;
        page.setData({src:tempPath});
      }
    })
  }
})
```

界面效果如图4.5所示。

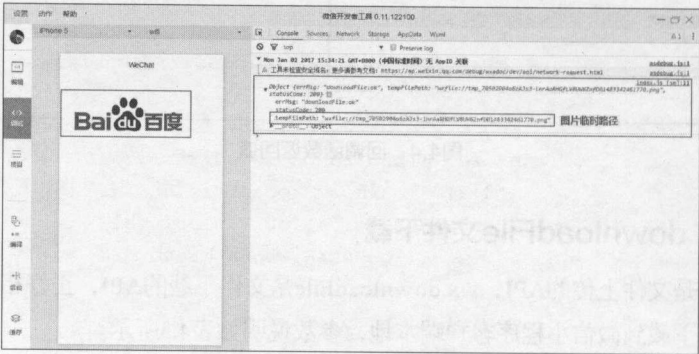


图4.5 下载图片

`wx.downloadFile` 文件下载最大并发限制是10个，默认超时时间和最大超时时间都是 60s，网络请求的 `referer` 是不可以设置的，格式固定为 `https://servicewechat.com/{Appid}/{version}/page-frame.html`，其中，`{Appid}` 为小程序的 Appid，`{version}` 为小程序的版本号，版本号为 0 表示为开发版。

4.3 WebSocket会话API

WebSocket会话用来创建一个会话连接，创建完会话连接后可以进行通信，如同微信聊天和QQ聊天一样。它会用到以下7个API的使用。

- 1 `wx.connectSocket(OBJECT)` 创建一个会话连接。
- 2 `wx.onSocketOpen(CALLBACK)` 监听WebSocket连接打开事件。
- 3 `wx.onSocketError(CALLBACK)` 监听WebSocket错误。
- 4 `wx.sendSocketMessage(OBJECT)` 发送数据。
- 5 `wx.onSocketMessage(CALLBACK)` 监听WebSocket接受到服务器的消息事件。
- 6 `wx.closeSocket()` 关闭WebSocket连接。
- 7 `wx.onSocketClose(CALLBACK)` 监听WebSocket关闭。

`wx.connectSocket(object)` 参数说明如表4.4所示。

表4.4 `wx.connectSocket` 参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|---|
| url | String | 是 | 开发者服务器 url |
| data | Object | 否 | 请求的数据 |
| header | Object | 否 | HTTP 请求 Header，header 中不能设置 Referer |
| method | String | 否 | 默认是GET，有效值为：OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

一个微信小程序同时只能有一个 WebSocket 连接，如果当前已存在一个 WebSocket 连接，会自动关闭该连接，并重新创建一个 WebSocket 连接。

`wx.sendSocketMessage(object)` 参数说明如表4.5所示。

表4.5 `wx.sendSocketMessage` 参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|--------------------|----|--------------------------|
| data | String/ArrayBuffer | 否 | 请求的数据 |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

下面，我们来演示WebSocket会话通信的使用。

1 在WXML文件里进行界面布局设计，创建一个连接按钮、发送消息以及接受消息，具体代码如下。

```
<button type="default" bindtap="createConn"> 创建连接 </button>
<view style="display:flex;flex-direction:row;margin:10px;">
<input type="text" name="msg" bindblur="getMsg" style="width:200px;border:1px solid #cccccc;"/>
<button type="primary" size="mini" bindtap="send"> 发送消息 </button>
</view>
<view style="height:200px;">
<view style="font-weight:bold;"> 客户端发送的消息 </view>
<block wx:for="{{sendMsg}}" wx:for-item="item1">
<view style="color:green">{{item1}}</view>
</block>
</view>

<view style="height:200px;">
<view style="font-weight:bold;"> 服务端返回的消息 </view>
<block wx:for="{{resData}}" wx:for-item="item2">
<view style="color:red">{{item2}}</view>
</block>
</view>
<view style="margin:10px;">{{content}}</view>
<button type="default" bindtap="closeConn"> 关闭连接 </button>
```

界面效果如图4.6所示。

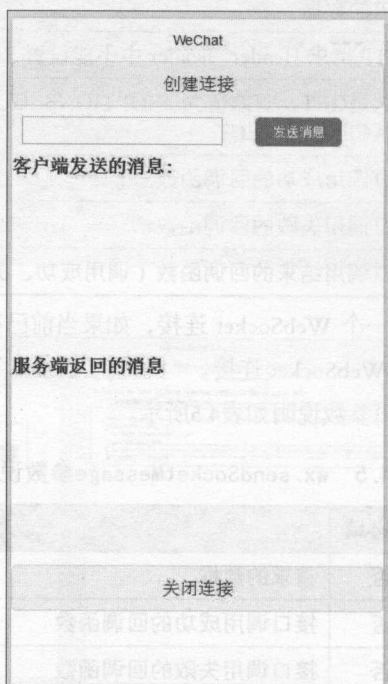


图4.6 WebSocket会话布局

2 在js文件里，利用WebSocket提供API：wx.connectSocket创建一个会话连接API，wx.onSocketOpen监听连接成功，wx.onSocketError监听连接打开失败，wx.sendSocketMessage显示发送消息，wx.closeSocket用来关闭连接，具体代码如下。

```
Page({
  data: {
    msg: '',
    sendMsg: [],
    socketOpen: false,
    resData: []
  },
  createConn: function() {
    var page = this;
    wx.connectSocket({
      url: 'ws://localhost:8555/wxApp/getServer',
      data: {
        x: '',
        y: ''
      },
      header: {
        'content-type': 'Application/json'
      },
      method: "GET"
    });
    wx.onSocketOpen(function(res) {
      console.log(res);
      page.setData({socketOpen:true});
      console.log('WebSocket 连接已打开! ');
    });
    wx.onSocketError(function(res) {
      console.log('WebSocket 连接打开失败，请检查! ');
    });
  },
  send: function(e) {
    if (this.data.socketOpen) {
      console.log(this.data.socketOpen);
      wx.sendSocketMessage({
        data: this.data.msg
      });
      var sendMsg = this.data.sendMsg;
      sendMsg.push(this.data.msg);
      this.setData({sendMsg:sendMsg});
      var page = this;
      wx.onSocketMessage(function(res) {
        var resData = page.data.resData;
        resData.push(res.data);
        page.setData({resData:resData});
        console.log(resData);
        console.log('收到服务器内容: ' + res.data)
      })
    }
  }
})
```

```

        } else {
            console.log('WebSocket 连接打开失败，请检查！ ');
        }
    },
    closeConn:function(e){
        wx.closeSocket();
        wx.onSocketClose(function(res) {
            console.log('WebSocket 已关闭! ')
        });
    },
    getMsg:function(e){
        var page = this;
        page.setData({msg:e.detail.value});
    }
})

```

3 在服务器端用Java语言代码编写WebSocket会话代码，具体代码如下。

```

package com.xiaogang.App.servlet;

import java.io.IOException;

import javax.websocket.OnClose;
import javax.websocket.OnError;
import javax.websocket.OnMessage;
import javax.websocket.OnOpen;
import javax.websocket.RemoteEndpoint;
import javax.websocket.Session;
import javax.websocket.server.ServerEndpoint;

@ServerEndpoint("/getServer")
public class WebSocketServer{

    @OnOpen
    public void onOpen(Session session) {
        System.out.println("sessionId="+session.getId());
        final RemoteEndpoint.Basic basic = session.getBasicRemote();

        try {
            basic.sendText(" 会话建立成功!!! ");
        } catch (IOException e) {
            e.printStackTrace();
        }

        Thread t1=new Thread(new Runnable() {

            @Override
            public void run() {
                try {
                    Thread.currentThread();
                    Thread.sleep(8000);

```



```

        basic.sendText("server get you a msg:
what your name?");

        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    });
    t1.start();
}
/**
 * 收到客户端消息时触发
 * @param relationId
 * @param userCode
 * @param message
 * @return
 */
@OnMessage
public String onMessage(Session session,String message) {
    System.out.println("pathParams:"+session.getPathParameters());
    System.out.println("requestParams"+session.getRequestParameterMap());
    return "Got you msg !"+message;
}

/**
 * 异常时触发
 * @param relationId
 * @param userCode
 * @param session
 */
@OnError
public void onError(Throwable throwable,Session session) {
    System.out.println("pathParams:"+session.getPathParameters());
    System.out.println("requestParams"+session.getRequestParameterMap());
    System.out.print("onError"+throwable.toString());
}

/**
 * 关闭连接时触发
 * @param relationId
 * @param userCode
 * @param session
 */
@OnClose
public void onClose(Session session) {
    System.out.println("pathParams:"+session.getPathParameters());
    System.out.println("requestParams"+session.getRequestParameterMap());
    System.out.print("onClose ");
}
}

```


4 微信小程序客户端和服务端代码写完之后，就可以单击“创建连接”按钮，创建一个会话连接，同时可以发送消息给服务端，并接收服务端传递过来的消息，最后单击“关闭连接”按钮，如图4.7所示。

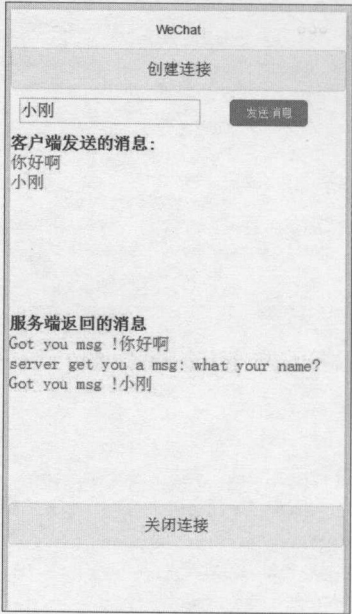


图4.7 会话聊天

4.4 图片处理API

微信小程序针对图片处理提供了3个API：`wx.chooseImage(OBJECT)`选择图片API、`wx.previewImage(OBJECT)`预览图片API、`wx.getImageInfo(OBJECT)`获得图片信息API。

4.4.1 wx.chooseImage(OBJECT)选择图片

`wx.chooseImage`选择图片API可以从本地相册选择图片或使用相机拍照来选择图片，参数说明如表4.6所示。

表4.6 wx.chooseImage参数说明

| 属性 | 类型 | 必填 | 说明 |
|------------|-------------|----|-----------------------------------|
| count | Number | 否 | 最多可以选择的图片张数，默认为9 |
| sizeType | StringArray | 否 | original 原图，compressed 压缩图，默认二者都有 |
| sourceType | StringArray | 否 | album 从相册选图，camera 使用相机，默认二者都有 |
| success | Function | 否 | 成功则返回图片的本地文件路径列表 tempFilePaths |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

`wx.chooseImage`选择图片API，通过`count`属性可以设置每次最多选择的图片数量，通过`sizeType`属性可以设置显示`original`原图或者`compressed`压缩图，通过`sourceType`属性可以设置`album`相册选图、`camera`使用相机选图或者两者都可以。

示例代码如下。

```
Page({
  onLoad:function(){
    wx.chooseImage({
      count: 9, // 默认为9
      sizeType: ['original', 'compressed'], // 可以指定是原图还是压缩图，默认二者都有
      sourceType: ['album', 'camera'], // 可以指定来源是相册还是相机，默认二者都有
      success: function (res) {
        // 返回选定照片的本地文件路径列表，tempFilePath 可以作为 img 标签的 src 属性显示图片
        var tempFilePaths = res.tempFilePaths
      }
    })
  }
})
```

4.4.2 wx.previewImage(OBJECT)预览图片

`wx.previewImage`预览图片API可以用来预览多张图片以及设置默认显示的图片，参数说明如表4.7所示。

表4.7 wx.previewImage参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|-------------|----|----------------------------|
| current | String | 否 | 当前显示图片的链接，不填则默认为 urls 的第一张 |
| urls | StringArray | 是 | 需要预览的图片链接列表 |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
Page({
  onLoad:function(){
    wx.previewImage({
      current: 'http://img02.tooopen.com/images/20150928/tooopen_sy_143912755726.jpg',
      // 当前显示图片的 http 链接
      urls: [
        "http://img02.tooopen.com/images/20150928/tooopen_sy_143912755726.jpg",
        "http://img06.tooopen.com/images/20160818/tooopen_sy_175866434296.jpg",
        "http://img06.tooopen.com/images/20160818/tooopen_sy_175833047715.jpg"
      ] // 需要预览的图片 http 链接列表
    })
  }
})
```

界面效果如图4.8和图4.9所示。

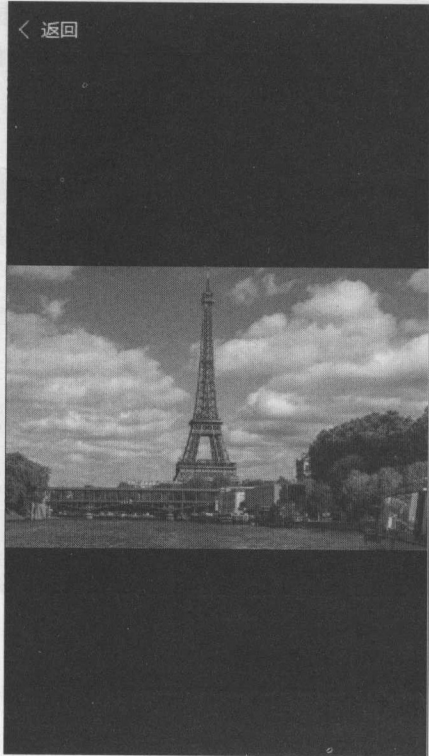


图4.8 预览一

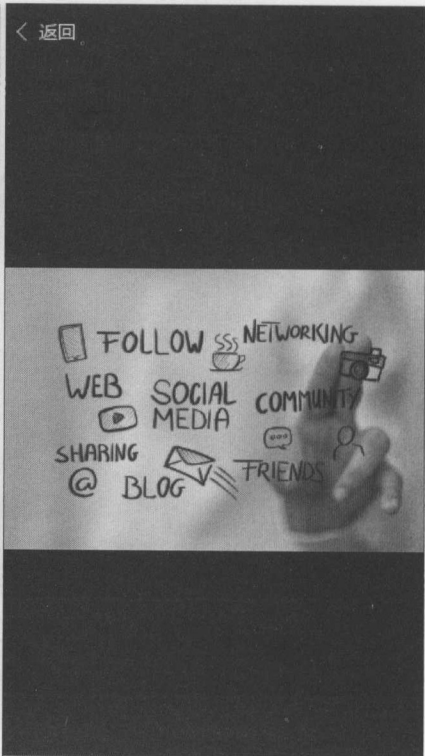


图4.9 预览二

4.4.3 wx.getImageInfo(OBJECT)获得图片信息

wx.getImageInfo用来获得图片信息，包括图片的宽度、图片的高度以及图片返回的图片路径，参数说明如表4.8所示。

表4.8 wx.getImageInfo参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|------------------------------------|
| src | String | 是 | 图片的路径，可以是相对路径、临时文件路径、存储文件路径、网络图片路径 |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.9所示。

表4.9 success返回参数说明

| 参数 | 类型 | 说明 |
|--------|--------|------------|
| width | Number | 图片宽度，单位为px |
| height | Number | 图片高度，单位为px |
| path | String | 返回图片的本地路径 |

示例代码如下。

```
Page({
  onLoad: function() {
    wx.getImageInfo({
      src: 'http://img02.tooopen.com/images/20150928/tooopen_sy_143912755726.jpg',
      success: function (res) {
        console.log(" 图片宽度 =" + res.width);
        console.log(" 图片高度 =" + res.height);
        console.log(" 图片返回路径 =" + res.path);
      }
    })
  }
})
```

打印信息如图4.10所示。



图4.10 图片信息

4.5 文件操作API

微信小程序针对文件操作提供了5个API：`wx.saveFile`将文件保存到本地、`wx.getSavedFileList`获取本地已保存的文件列表、`wx.getSavedFileInfo`获取本地文件信息、`wx.getSavedFileInfo`删除本地文件、`wx.openDocument`打开文档。

4.5.1 `wx.saveFile`保存文件到本地

`wx.saveFile(object)`可以根据文件的临时路径，将文件保存到本地，下次启动微信小程序的时候，仍然可以获取到该文件；如果是临时路径，下次启动微信小程序的时候，就无法获取到该文件。本地文件存储的大小限制为10M。参数说明如表4.10所示。

表4.10 `wx.saveFile`参数说明

| 属性 | 类型 | 必填 | 说明 |
|--------------|----------|----|--|
| tempFilePath | String | 是 | 需要保存的文件的临时路径 |
| success | Function | 否 | 返回文件的保存路径， <code>res = {savedFilePath: '文件的保存路径'}</code> |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
Page({
  onLoad:function(){
    wx.getImageInfo({
      src: 'http://img02.tooopen.com/images/20150928/tooopen_sy_143912755726.jpg',
      success: function (res) {
        var path = res.path;
        console.log(" 临时文件路径="+path);
        wx.saveFile({
          tempFilePath: path,
          success: function(res){
            var savedFilePath = res.savedFilePath;
            console.log(" 本地文件路径="+savedFilePath);
          }
        })
      }
    })
  }
})
```

将文件保存到本地后，会返回一个savedFilePath本地文件存储路径，根据这个路径就可以访问或者使用该文件，同时在微信小程序下次启动的时候，这个本地文件仍然存在。

4.5.2 wx.getSavedFileList获取本地文件列表

通过wx.saveFile可以将临时文件保存到本地，成为本地文件，可以通过wx.getSavedFileList来获取本地文件列表，获取到wx.saveFile保存的文件，参数说明如表4.11所示。

表4.11 wx.getSavedFileList参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------------|
| success | Function | 否 | 接口调用成功的回调函数，返回结果见success返回参数说明 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.12所示。

表4.12 success返回参数说明

| 参数 | 类型 | 说明 |
|----------|--------------|--------|
| errMsg | String | 接口调用结果 |
| fileList | Object Array | 文件列表 |

fileList中的项目说明如表4.13所示。

表4.13 fileList说明

| 键 | 类型 | 说明 |
|----------|--------|---------|
| filePath | String | 文件的本地路径 |

续表

| 键 | 类型 | 说明 |
|------------|--------|---|
| createTime | Number | 文件保存时的时间戳，从1970/01/01 08:00:00 到当前时间的秒数 |
| size | Number | 文件大小，单位为B |

示例代码如下。

```
Page({
  onLoad: function() {
    wx.getSavedFileList({
      success: function(res) {
        var fileList = res.fileList;
        console.log(fileList)
        for(var i=0;i<fileList.length;i++){
          var file = fileList[i];
          console.log(" 第 "+(i+1)+" 个文件:");
          console.log(" 文件创建时间="+file.createTime);
          console.log(" 文件大小="+file.size);
          console.log(" 文件本地路径="+file.filePath);
        }
      }
    })
  }
})
```

打印信息如图4.11所示。

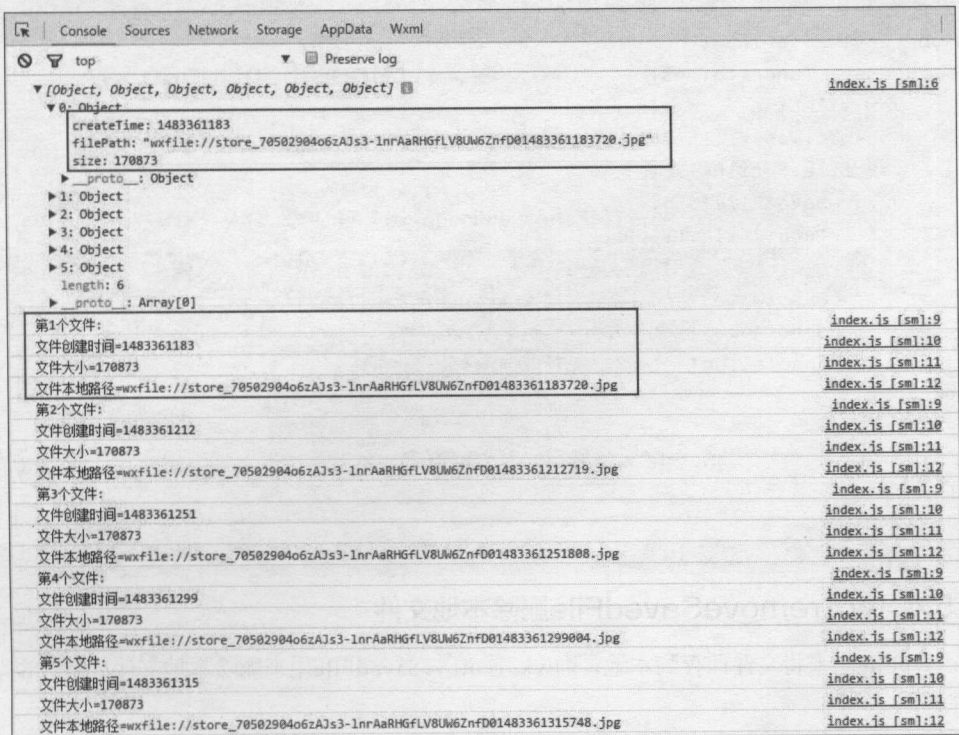


图4.11 本地文件信息

4.5.3 wx.getSavedFileInfo获取本地文件信息

wx.getSavedFileInfo获取本地指定路径的文件信息，包括文件的创建时间、文件的大小以及接口调用结果。wx.getSavedFileInfo参数说明如表4.14所示。

表4.14 wx.getSavedFileInfo参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------------|
| filePath | String | 是 | 文件路径 |
| success | Function | 否 | 接口调用成功的回调函数，返回结果见success返回参数说明 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.15所示。

表4.15 success返回参数说明

| 参数 | 类型 | 说明 |
|------------|--------|---|
| errMsg | String | 接口调用结果 |
| size | Number | 文件大小，单位为B |
| createTime | Number | 文件保存时的时间戳，从1970/01/01 08:00:00 到当前时间的秒数 |

示例代码如下。

```
Page({
  onLoad:function(){
    wx.getSavedFileList({
      success: function(res) {
        var fileList = res.fileList;
        console.log(fileList)
        var file = fileList[0];
        wx.getSavedFileInfo({
          filePath: file.filePath,
          success: function(res){
            console.log(" 文件创建时间 =" +res.createTime);
            console.log(" 文件大小 =" +res.size);
            console.log(" 文件本地路径 =" +res.errMsg);
          }
        })
      }
    })
  }
})
```

4.5.4 wx.removeSavedFile删除本地文件

wx.saveFile用来将文件保存到本地，而wx.removeSavedFile用来删除本地文件，参数说明如表4.16所示。

表4.16 wx.removeSavedFile参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| filePath | String | 是 | 需要删除的文件路径 |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
Page({
  onLoad:function(){
    wx.getSavedFileList({
      success: function(res) {
        var fileList = res.fileList;
        console.log(fileList)
        var file = fileList[0];
        wx.removeSavedFile({
          filePath: file.filePath,
          complete: function(res) {
            console.log(res)
          }
        })
      }
    })
  }
})
```

4.5.5 wx.openDocument打开文档

wx.openDocument可以打开doc、xls、ppt、pdf、docx、xlsx、pptx等多种格式的文档，参数说明如表4.17所示。

表4.17 wx.openDocument参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| filePath | String | 是 | 文件路径，可通过 downFile 获得 |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
Page({
  onLoad:function(){
    wx.downloadFile({
      url: 'http://www.crcc.cn/portals/0/word/ 应聘材料样本.doc',
      success: function (res) {
        var filePath = res.tempFilePath
        wx.openDocument({
```

```
        filePath: filePath,
        success: function (res) {
            console.log(' 打开文档成功 ')
        }
    })
}
})
}
```

4.6 数据缓存API

微信小程序数据缓存API用来处理数据缓存信息，可以将数据缓存到本地、获取到本地缓存数据、移除缓存数据以及清理缓存数据。常用的数据缓存API有以下几种。

- 1** wx.setStorage(OBJECT) 异步方式将数据存储在本地图存中指定的 key 中。
- 2** wx.setStorageSync(KEY,DATA)同步方式将数据存储在本地图存中指定的 key 中。
- 3** wx.getStorage(OBJECT)异步方式从本地图存中获取指定 key 对应的内容。
- 4** wx.getStorageSync(KEY)同步方式从本地图存中获取指定 key 对应的内容。
- 5** wx.getStorageInfo(OBJECT)异步方式获取当前storage的相关信息。
- 6** wx.getStorageInfoSync(OBJECT)同步方式获取当前storage的相关信息。
- 7** wx.removeStorage(OBJECT) 异步方式从本地图存中移除指定的key。
- 8** wx.removeStorageSync(KEY) 同步方式从本地图存中移除指定的key。
- 9** wx.clearStorage()异步方式清理本地数据缓存。
- 10** wx.clearStorageSync()同步方式清理本地数据缓存。

4.6.1 数据缓存到本地

微信小程序数据缓存到本地提供了两种方式：一种是wx.setStorage(OBJECT)异步方式将数据存储到本地图存中指定的 key 中；另一种是wx.setStorageSync(KEY,DATA)同步方式将数据存储到本地图存中指定的 key 中，本地缓存最大为10MB。

1. wx.setStorage(OBJECT)

异步方式将数据存储到本地图存中指定的 key 中，会覆盖掉原来该 key 对应的内容，参数说明如表4.18所示。

表4.18 wx.setStorage参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|---------------|----|--------------------------|
| key | String | 是 | 本地缓存中指定的 key |
| data | Object/String | 是 | 需要存储的内容 |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

如果我们想把用户信息缓存到本地，示例代码如下。

```
Page({
  onLoad:function(){
    var user = this.getUserInfo();
    console.log(user);
    wx.setStorage({
      key: 'user',
      data: user,
      success: function(res){
        console.log(res);
      }
    })
  },
  getUserInfo:function(){
    var user = new Object();
    user.name = 'xiaogang';
    user.sex = '男';
    user.age = 30;
    user.address='北京市';
    return user;
  }
})
```

在Storage里可以查看到缓存的数据，如图4.12所示。

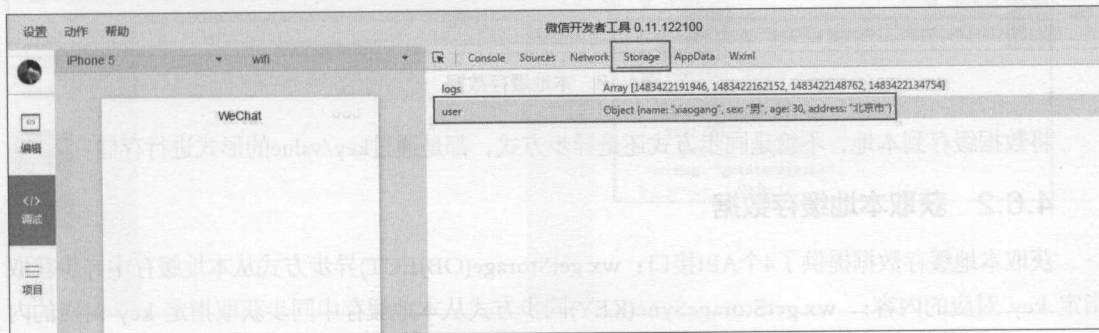


图4.12 本地缓存数据

2. wx.setStorageSync(KEY,DATA)

同步方式将数据存储到本地指定的key中，会覆盖掉原来该 key 对应的内容，相比于异步缓存数据，它更简练一些，参数说明如表4.19所示。

表4.19 wx.setStorageSync参数说明

| 属性 | 类型 | 必填 | 说明 |
|------|---------------|----|--------------|
| key | String | 是 | 本地缓存中指定的 key |
| data | Object/String | 是 | 需要存储的内容 |

示例代码如下。

```
Page({
  onLoad:function(){
    var userSync = this.getUserInfo();
    // 同步方式将数据存储到本地
    wx.setStorageSync('userSync', userSync)
  },
  getUserInfo:function(){
    var user = new Object();
    user.name = 'xiaogang';
    user.sex = '男';
    user.age = 30;
    user.address='北京市';
    return user;
  }
})
```

在Storage里可以查看到缓存的数据，如图4.13所示。

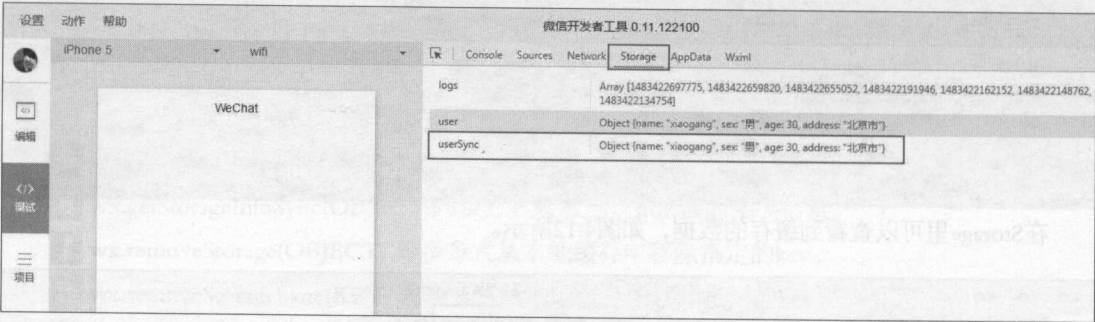


图4.13 本地缓存数据

将数据缓存到本地，不管是同步方式还是异步方式，都是通过key/value的形式进行存储。

4.6.2 获取本地缓存数据

获取本地缓存数据提供了4个API接口：wx.getStorage(OBJECT)异步方式从本地缓存中异步获取指定 key 对应的内容；wx.getStorageSync(KEY)同步方式从本地缓存中同步获取指定 key 对应的内容；wx.getStorageInfo(OBJECT)异步方式获取当前storage的相关信息；wx.getStorageInfoSync同步方式获取当前storage的相关信息。前两个API接口是通过指定的key值获得缓存数据，而后两个API接口是获取当前storage的相关信息。

1. wx.getStorage (OBJECT)

wx.getStorage(OBJECT)使用异步方式从本地缓存中获取指定 key 对应的内容。参数说明如表4.20所示。

表4.20 wx.getStorage参数说明

| 属性 | 类型 | 必填 | 说明 |
|---------|----------|----|----------------------------------|
| key | String | 是 | 本地缓存中指定的 key |
| success | Function | 否 | 接口调用的回调函数，res = {data: key对应的内容} |

续表

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

在前面章节中，使用wx.setStorage(OBJECT)将user异步方式保存到本地，下面使用wx.getStorage(OBJECT)来获取本地数据，具体代码如下。

```
Page({
  onLoad:function(){
    // 异步方式获取本地数据
    wx.getStorage({
      key: 'user',
      success: function(res){
        console.log(res);
      }
    })
  }
})
```

获取到的本地数据如图4.14所示。

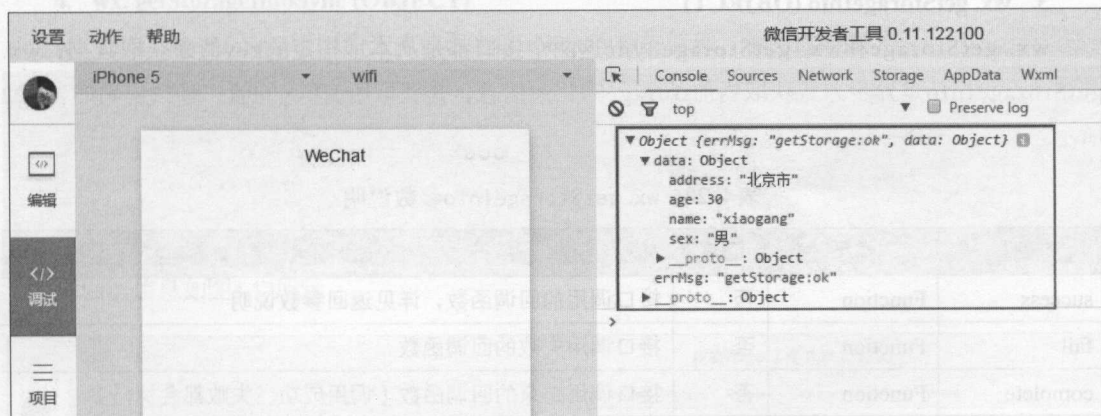


图4.14 异步获取本地数据

2. wx.getStorageSync(OBJECT)

wx.getStorageSync(OBJECT)是一个同步的接口，用来从本地缓存中同步获取指定 key 对应的内容。它只有一个参数，如表4.21所示。

表4.21 wx.getStorageSync参数说明

| 属性 | 类型 | 必填 | 说明 |
|-----|--------|----|--------------|
| key | String | 是 | 本地缓存中指定的 key |

示例代码如下。

```
Page({
  onLoad:function(){
```



```
// 同步方式获取本地数据
var userSync = wx.getStorageSync('userSync');
console.log(userSync);
}
))
```

获取到的本地数据如图4.15所示。

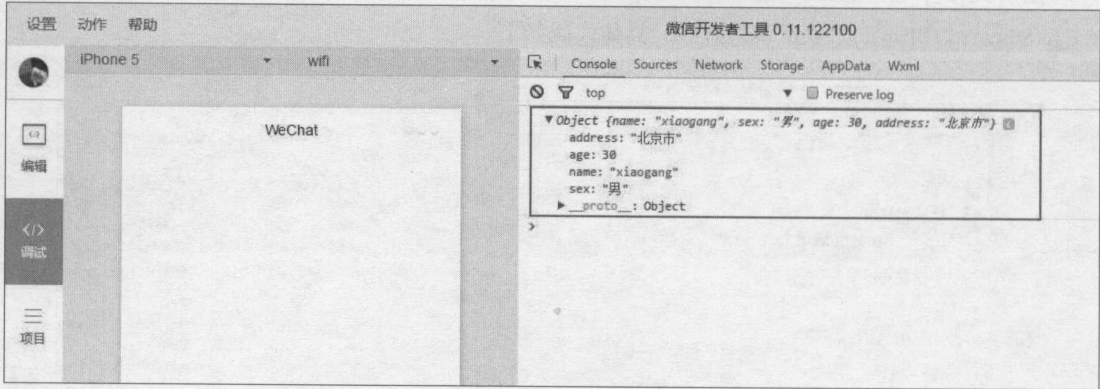


图4.15 同步获取本地数据

3. wx.getStorageInfo (OBJECT)

wx.getStorage和wx.getStorageSync这两个接口都是从本地指定的key值来获取数据，wx.getStorageInfo是异步方式获取当前storage的相关信息，是获取所有key的值，参数说明如表4.22所示。

表4.22 wx.getStorageInfo参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| success | Function | 否 | 接口调用的回调函数，详见返回参数说明 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.23所示。

表4.23 success返回参数说明

| 参数 | 类型 | 说明 |
|-------------|--------------|------------------|
| keys | String Array | 当前storage中所有的key |
| currentSize | Number | 当前占用的空间大小, 单位为kb |
| limitSize | Number | 限制的空间大小, 单位为kb |

示例代码如下。

```
Page({
  onLoad:function(){
    wx.getStorageInfo({
      success: function(res){
```

```
    console.log(res);  
  }  
})  
}
```

返回值信息如图4.16所示。

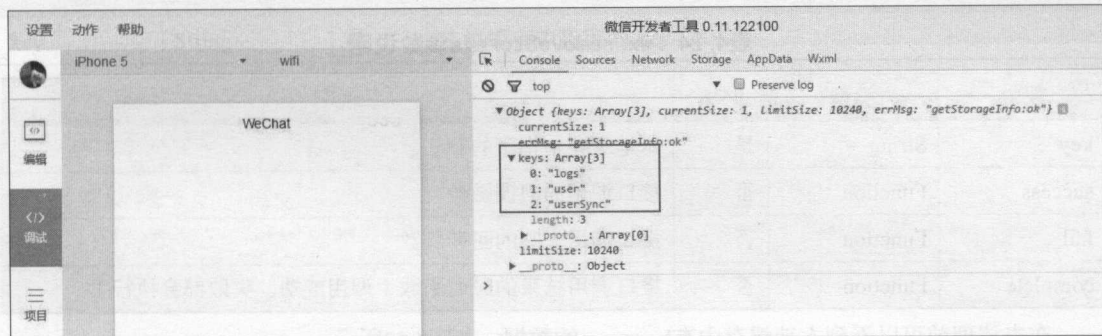


图4.16 异步获取本地key数据

获取到本地所有的key值后，根据这个key值就可以查找到相应的数据。

4. wx.getStorageInfoSync (OBJECT)

wx.getStorageInfoSync以同步方式获取当前storage的相关信息，示例代码如下。

```
Page({  
  onLoad:function(){  
    var storage = wx.getStorageInfoSync();  
    console.log(storage);  
  }  
})
```

返回值信息如图4.17所示。

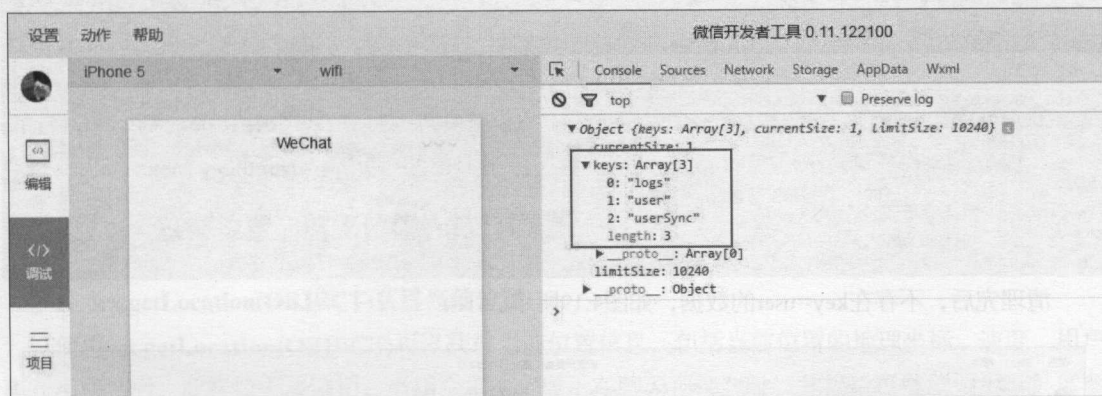


图4.17 同步获取本地key数据

它和wx.getStorageInfo异步获取storage返回数据一样，都是返回所有的key值，然后根据key值再查找完整的数据。

4.6.3 移除和清理本地缓存数据

wx.removeStorage(OBJECT)、wx.removeStorageSync(KEY)用来从本地缓存中移除指定 key；wx.clearStorage()、wx.clearStorageSync()用来清理本地数据缓存。

1.wx.removeStorage(OBJECT)

wx.removeStorage(OBJECT)用来异步从本地缓存中移除指定的key，参数说明如表4.24所示。

表4.24 wx.removeStorage参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| key | String | 是 | 本地缓存中指定的 key |
| success | Function | 否 | 接口调用的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

在未清理前可以看到本地缓存中有key=user的数据，如图4.18所示。



图4.18 user缓存数据

下面，我们从本地缓存中清理key=user的数据，具体代码如下。

```
Page({
  onLoad:function(){
    // 异步移除 key=user 数据
    wx.removeStorage({
      key: 'user',
      success: function(res){
        console.log(res);
      },
    })
  }
})
```

清理完后，不存在key=user的数据，如图4.19所示。



图4.19 清理user缓存数据

2. wx.removeStorageSync(KEY)

wx.removeStorageSync (OBJECT)用来同步从本地缓存中移除指定的key，它的效果和wx.removeStorage一样，参数说明如表4.25所示。

表4.25 wx.removeStorageSync参数说明

| 属性 | 类型 | 必填 | 说明 |
|-----|--------|----|--------------|
| key | String | 是 | 本地缓存中指定的 key |

示例代码如下。

```
Page({
  onLoad:function(){
    // 同步移除 key=userSync 数据
    wx.removeStorageSync('userSync');
  }
})
```

3. wx.clearStorage ()、wx.clearStorageSync ()

wx.clearStorage ()、wx.clearStorageSync ()用来清理本地所有缓存数据，前者是异步清理缓存数据，后者是同步清理缓存数据。

示例代码如下。

```
wx.clearStorage()

try {
  wx.clearStorageSync()
} catch(e) {
}
```

4.7 位置信息API

微信小程序针对位置新提供了4个API接口：wx.getLocation(OBJECT)获得当前位置信息、wx.chooseLocation(OBJECT)打开地图选择位置；wx.openLocation(OBJECT) 使用微信内置地图查看位置；wx.createMapContext(mapId)地图组件控制创建并返回 map 上下文 mapContext 对象。

4.7.1 获得位置、选择位置、打开位置

1. wx.getLocation(OBJECT)获得当前位置

使用wx.getLocation(OBJECT)可以获得当前位置信息，包括当前位置的地理坐标、速度，用户离开小程序后，此接口无法调用；当用户单击“显示在聊天顶部”时，此接口可继续调用。具体参数如表4.26所示。

表4.26 wx.getLocation参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|---|
| type | String | 否 | 默认为 wgs84 返回 gps 坐标，gcj02 返回可用于wx.openLocation的坐标 |
| success | Function | 是 | 接口调用的回调函数，详见返回参数说明 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.27所示。

表4.27 success返回参数说明

| 参数 | 说明 |
|-----------|---------------------------|
| latitude | 纬度，浮点数，范围为-90~90，负数表示南纬 |
| longitude | 经度，浮点数，范围为-180~180，负数表示西经 |
| speed | 速度，浮点数，单位为m/s |
| accuracy | 位置的精确度 |

示例代码如下。

```
Page({
  onLoad: function() {
    wx.getLocation({
      type: 'wgs84',
      success: function(res) {
        var latitude = res.latitude;
        console.log(" 纬度="+latitude);
        var longitude = res.longitude;
        console.log(" 经度="+longitude);
        var speed = res.speed;
        console.log(" 速度="+speed);
        var accuracy = res.accuracy;
        console.log(" 精确度="+accuracy);
      }
    })
  }
})
```

2.wx.chooseLocation(OBJECT)选择位置

使用wx.chooseLocation打开地图来选择位置，具体参数说明如表4.28所示。

表4.28 wx.chooseLocation参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| success | Function | 是 | 接口调用的回调函数，详见返回参数说明 |
| cancel | Function | 否 | 用户取消时调用 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.29所示。

表4.29 success返回参数说明

| 参数 | 说明 |
|-----------|---------------------------|
| latitude | 纬度，浮点数，范围为-90~90，负数表示南纬 |
| longitude | 经度，浮点数，范围为-180~180，负数表示西经 |
| name | 位置名称 |
| address | 详细地址 |

示例代码如下。

```
Page({
  onLoad:function(){
    wx.chooseLocation({
      success: function(res){
        console.log(res);
      }
    })
  }
})
```

在地图中选择位置，如图4.20所示。

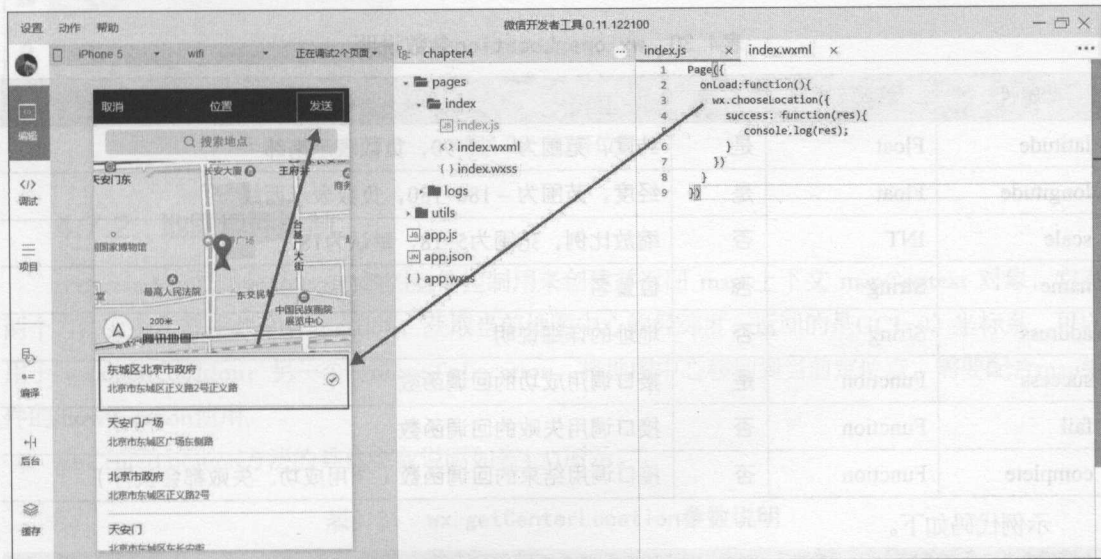


图4.20 选择位置

单击“发送”按钮，就可以把选择的位置信息打印出来，如图4.21所示。

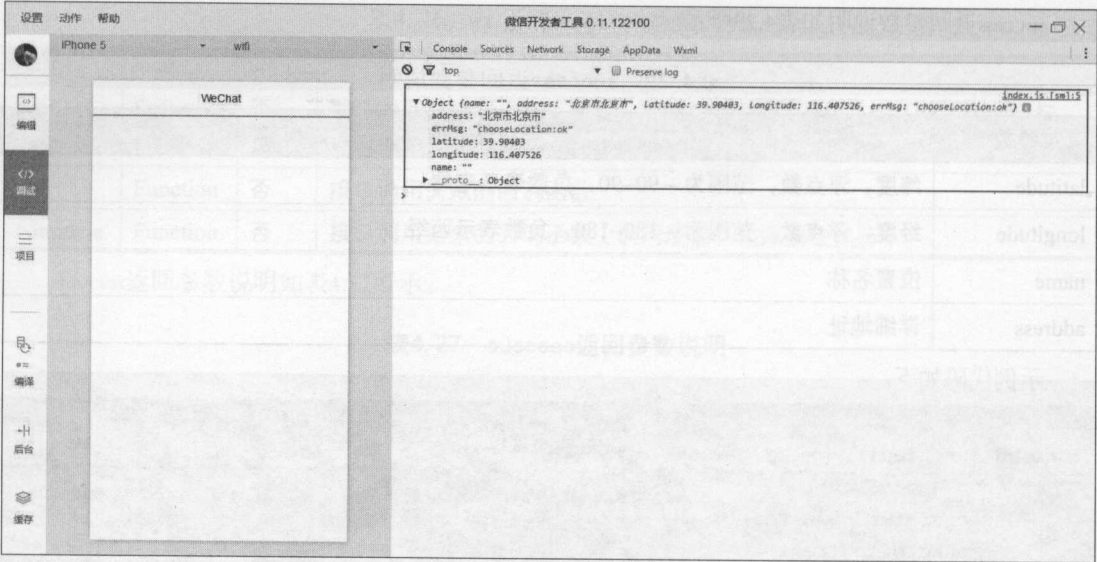


图4.21 位置信息

3. wx.openLocation(OBJECT)打开位置

使用wx.openLocation(OBJECT)接口可以使用微信内置地图查看位置，具体参数说明如表4.30所示。

表4.30 wx.openLocation参数说明

| 属性 | 类型 | 必填 | 说明 |
|-----------|----------|----|--------------------------|
| latitude | Float | 是 | 纬度，范围为-90~90，负数表示南纬 |
| longitude | Float | 是 | 经度，范围为-180~180，负数表示西经 |
| scale | INT | 否 | 缩放比例，范围为5~18，默认为18 |
| name | String | 否 | 位置名 |
| address | String | 否 | 地址的详细说明 |
| success | Function | 是 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
Page({
  onLoad:function(){
    wx.getLocation({
      type: 'gcj02', // 返回可以用于 wx.openLocation 的经纬度
      success: function(res) {
        var latitude = res.latitude
        var longitude = res.longitude
        wx.openLocation({
          latitude: latitude,
```

```

        longitude: longitude,
        scale: 28
    })
  }
})
}
})

```

界面效果如图4.22所示。

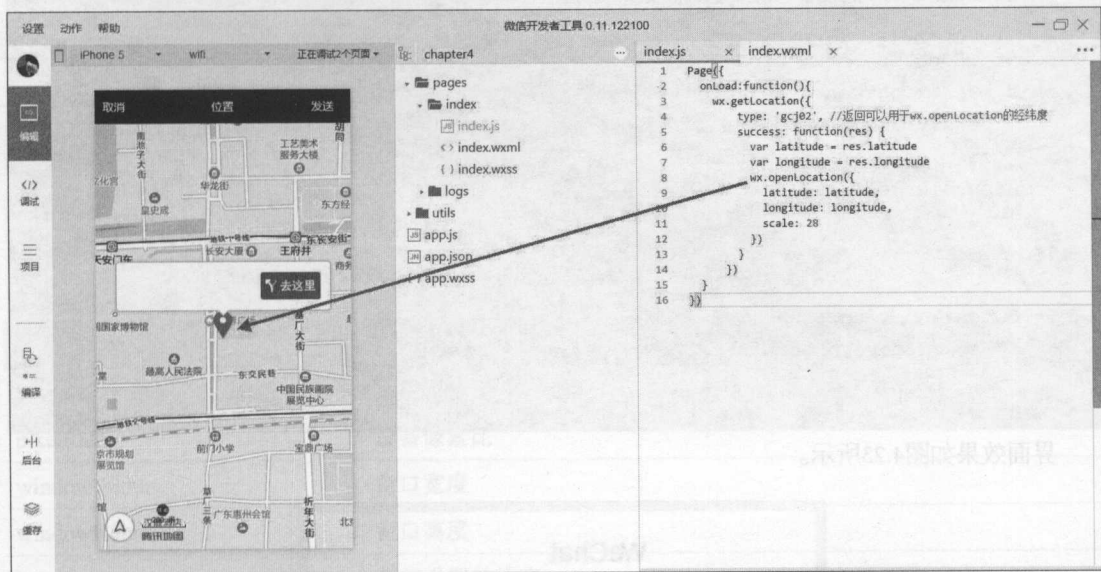


图4.22 打开位置

4.7.2 地图组件控制

`wx.createMapContext(mapId)`地图组件控制用来创建并返回 `map` 上下文 `mapContext` 对象，它有两个方法：一个是 `getCenterLocation`，获取当前地图中心的经纬度，返回的是GCJ-02 坐标系，可以用于 `wx.openLocation`；另一个是 `moveToLocation`，将地图中心移动到当前定位点，需要配合 `map` 组件的 `show-location` 使用。

`getCenterLocation`方法的具体参数说明如表4.31所示。

表4.31 `wx.getCenterLocation`参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|---|
| success | Function | 是 | 接口调用成功的回调函数， <code>res = { longitude: "经度", latitude: "纬度" }</code> |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```

<!-- map.wxml -->
<map id="myMap" show-location />

```

```
<button type="primary" bindtap="getCenterLocation"> 获取位置 </button>
<button type="primary" bindtap="moveToLocation"> 移动位置 </button>
```

```
// map.js
Page({
  onReady: function (e) {
    // 使用 wx.createMapContext 获取 map 上下文
    this.mapCtx = wx.createMapContext('myMap')
  },
  getCenterLocation: function () {
    this.mapCtx.getCenterLocation({
      success: function(res){
        console.log(res.longitude)
        console.log(res.latitude)
      }
    })
  },
  moveToLocation: function () {
    this.mapCtx.moveToLocation()
  }
})
```

界面效果如图4.23所示。



图4.23 地图组件控制界面效果

4.8 设备应用API

微信小程序针对设备应用6类API：获得系统信息、获取网络状态、重力感应、罗盘、拨打电话、扫码，这6类都是针对设备应用提供的相关接口。

4.8.1 获得系统信息

获得系统信息提供了两个API：一个是异步获取系统信息的`wx.getSystemInfo(OBJECT)`；另一个是同步获取系统信息的`wx.getSystemInfoSync()`。

1. `wx.getSystemInfo(OBJECT)`异步获取系统信息

`wx.getSystemInfo(OBJECT)`用来异步获取设备的系统信息，具体参数说明如表4.32所示。

表4.32 `wx.getSystemInfo`参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| success | Function | 是 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.33所示。

表4.33 success返回参数说明

| 参数 | 说明 |
|--------------|---------|
| model | 手机型号 |
| pixelRatio | 设备像素比 |
| windowWidth | 窗口宽度 |
| windowHeight | 窗口高度 |
| language | 微信设置的语言 |
| version | 微信版本号 |
| system | 操作系统版本 |
| platform | 客户端平台 |

示例代码如下。

```
Page({
  onLoad: function() {
    wx.getSystemInfo({
      success: function(res) {
        console.log(" 手机型号 =" + res.model)
        console.log(" 设备像素比 =" + res.pixelRatio)
        console.log(" 窗口宽度 =" + res.windowWidth)
        console.log(" 窗口高度 =" + res.windowHeight)
        console.log(" 微信设置的语言 =" + res.language)
        console.log(" 微信版本号 =" + res.version)
        console.log(" 操作系统版本 =" + res.system)
        console.log(" 客户端平台 =" + res.platform)
      }
    })
  }
})
```

2. wx.getSystemInfoSync()同步获取系统信息

wx.getSystemInfoSync用来同步获取系统信息，它是没有参数的，示例代码如下。

```
Page({
  onLoad: function () {
    try {
      var res = wx.getSystemInfoSync()
      console.log(" 手机型号 =" + res.model)
      console.log(" 设备像素比 =" + res.pixelRatio)
      console.log(" 窗口宽度 =" + res.windowWidth)
      console.log(" 窗口高度 =" + res.windowHeight)
      console.log(" 微信设置的语言 =" + res.language)
      console.log(" 微信版本号 =" + res.version)
      console.log(" 操作系统版本 =" + res.system)
      console.log(" 客户端平台 =" + res.platform)
    } catch (e) {
      // Do something when catch error
    }
  }
})
```

4.8.2 获取网络状态

微信小程序使用wx.getNetworkType(OBJECT)来获取网络类型，网络类型分为2g、3g、4g、wifi，具体参数如表4.34所示。

表4.34 wx.getNetworkType参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|---------------------------|
| success | Function | 是 | 接口调用成功，返回网络类型 networkType |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
Page({
  onLoad: function () {
    wx.getNetworkType({
      success: function (res) {
        // 返回网络类型 2g, 3g, 4g, wifi
        var networkType = res.networkType;
        console.log(" 网络类型 =" + networkType);
      }
    })
  }
})
```

4.8.3 重力感应

微信小程序使用wx.onAccelerometerChange(CALLBACK)来进行重力感应，监听重力感应数据，频率为5次/秒，具体参数说明如表4.35所示。

表4.35 wx.onAccelerometerChange参数说明

| 参数 | 类型 | 说明 |
|----|--------|-----|
| x | Number | X 轴 |
| y | Number | Y 轴 |
| z | Number | Z 轴 |

示例代码如下。

```
Page({
  onLoad: function () {
    wx.onAccelerometerChange(function(res) {
      console.log("X 轴 =" + res.x)
      console.log("Y 轴 =" + res.y)
      console.log("Z 轴 =" + res.z)
    })
  }
})
```

4.8.4 罗盘

微信小程序使用wx.onCompassChange(CALLBACK)来监听罗盘数据，频率为5次/秒，具体参数说明如表4.36所示。

表4.36 wx.onCompassChange参数说明

| 参数 | 类型 | 说明 |
|-----------|--------|---------|
| direction | Number | 面对的方向度数 |

示例代码如下。

```
Page({
  onLoad: function () {
    wx.onCompassChange(function (res) {
      console.log(" 面对的方向度数 =" + res.direction)
    })
  }
})
```

4.8.5 拨打电话

微信小程序使用wx.makePhoneCall(OBJECT)来拨打电话，具体参数说明如表4.37所示。

表4.37 wx.makePhoneCall参数说明

| 属性 | 类型 | 必填 | 说明 |
|-------------|----------|----|--------------------------|
| phoneNumber | String | 是 | 需要拨打的电话号码 |
| success | Function | 是 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
wx.makePhoneCall({
  phoneNumber: '13811112222'
})
```

4.8.6 扫码

微信小程序使用wx.scanCode(OBJECT)来调起客户端扫码界面，扫码成功后返回对应的结果，具体参数说明如表4.38所示。

表4.38 wx.scanCode参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| success | Function | 是 | 接口调用成功的回调函数，返回内容详见返回参数说明 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.39所示。

表4.39 success返回参数说明

| 参数 | 说明 |
|--------|------|
| result | 码的内容 |

示例代码如下。

```
wx.scanCode({
  success: (res) => {
    console.log(res)
  }
})
```

4.9 交互反馈API

微信小程序在用户操作过程中，提供了4种交互反馈API：wx.showToast(OBJECT) 显示消息提示框、wx.hideToast()隐藏消息提示框、wx.showModal(OBJECT) 显示模态弹窗、wx.showActionSheet(OBJECT)显示操作菜单。

4.9.1 消息提示框

消息提示框经常用来提交成功或者加载中的一种友好提示方式，如图4.24和图4.25所示。



图4.24 success类型

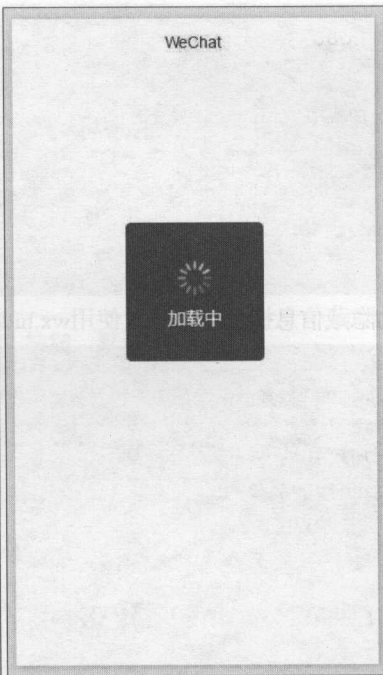


图4.25 loading类型

可以设置提示框的内容、类型、时间以及相应的事件，如果想显示消息提示框，可以使用 `wx.showToast(OBJECT)` 的API，它的具体参数说明如表4.40所示。

表4.40 wx.showToast参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|----------------------------------|
| title | String | 是 | 提示的内容 |
| icon | String | 否 | 图标，只支持“success”“loading” |
| duration | Number | 否 | 提示的延迟时间，单位为毫秒，默认为1 500，最大为10 000 |
| mask | Boolean | 否 | 是否显示透明蒙层，防止触摸穿透，默认为false |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
Page({
  onLoad: function () {
    wx.showToast({
      title: '成功',
      icon: 'success',
      duration: 2000
    })
  }
})
```

```
Page({
  onLoad: function () {
    wx.showToast({
      title: '加载中',
      icon: 'loading',
      duration: 2000
    })
  }
})
```

如果想手动隐藏信息提示框，可以使用wx.hideToast()接口，示例代码如下。

```
Page({
  onLoad: function () {
    wx.showToast({
      title: '加载中',
      icon: 'loading',
      duration: 10000
    })

    setTimeout(function () {
      wx.hideToast()
    }, 2000)
  }
})
```

4.9.2 模态弹窗

模态弹窗是对整个界面进行覆盖，防止用户对界面中的其他内容进行操作，如图4.26所示。

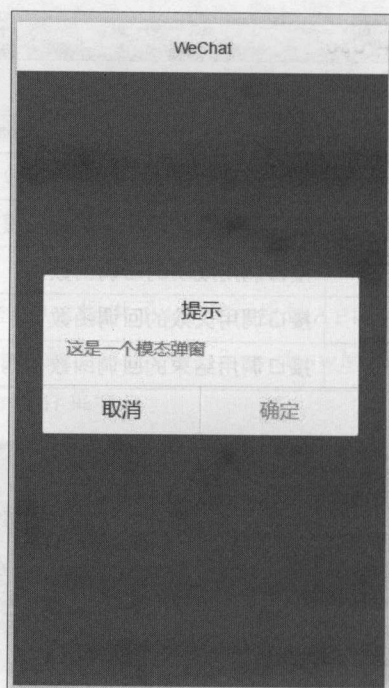


图4.26 模态弹窗

使用wx.showModal(OBJECT)显示模态弹窗，可以设置提示的标题、提示的内容、“取消”按钮和样式、“确定”按钮和样式以及一些绑定的事件，具体参数说明如表4.41所示。

表4.41 wx.showModal参数说明

| 属性 | 类型 | 必填 | 说明 |
|--------------|----------|----|--|
| title | String | 是 | 提示的标题 |
| content | String | 是 | 提示的内容 |
| showCancel | Boolean | 否 | 是否显示“取消”按钮，默认为 true |
| cancelText | String | 否 | “取消”按钮的文字，默认为“取消”，最多为 4 个字符 |
| cancelColor | HexColor | 否 | “取消”按钮的文字颜色，默认为 “#000000” |
| confirmText | String | 否 | “确定”按钮的文字，默认为“确定”，最多为 4 个字符 |
| confirmColor | HexColor | 否 | “确定”按钮的文字颜色，默认为 “#3CC51F” |
| success | Function | 否 | 接口调用成功的回调函数，返回res.confirm为true时，表示用户单击“确定”按钮 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
Page({
  onLoad: function () {
    wx.showModal({
      title: '提示',
      content: '这是一个模态弹窗',
      success: function (res) {
        if (res.confirm) {
          console.log('用户单击确定')
        }
      }
    })
  }
})
```

4.9.3 操作菜单

在App软件里，经常可以看到会从底部弹出很多选项供我们选择，也可以取消选择，如图4.27所示。

在微信小程序里，同样可以实现这样的效果，需要使用wx.showActionSheet(OBJECT)显示操作菜单这个API接口，具体参数说明如表4.42所示。

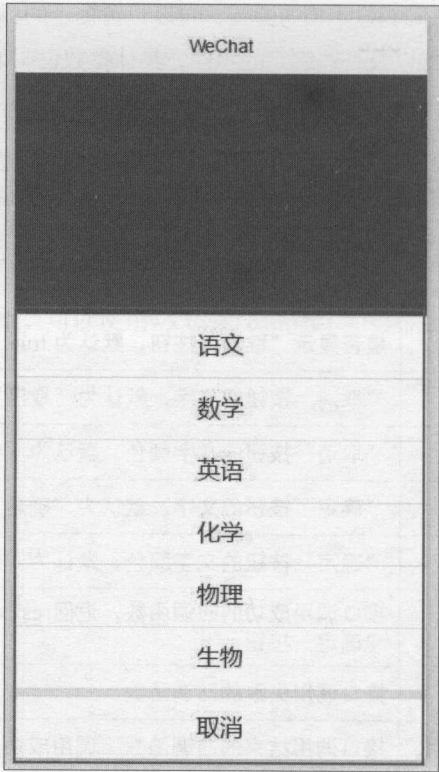


图4.27 操作菜单

表4.42 wx.showActionSheet参数说明

| 属性 | 类型 | 必填 | 说明 |
|-----------|--------------|----|--------------------------|
| itemList | String Array | 是 | 按钮的文字数组，数组长度最大为6个 |
| itemColor | HexColor | 否 | 按钮的文字颜色，默认为“#000000” |
| success | Function | 否 | 接口调用成功的回调函数，详见返回参数说明 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
Page({
  onLoad: function () {
    wx.showActionSheet({
      itemList: ['语文', '数学', '英语', '化学', '物理', '生物'],
      success: function (res) {
        if (!res.cancel) {
          console.log(res.tapIndex)
        }
      }
    })
  }
})
```

4.10 登录API

微信小程序的登录是必不可少的环节，它的登录可以简单理解为以下几个步骤。

- 1 使用wx.login获取code值。
- 2 拿到code值后再加上AppID、secret（在公众开发平台AppID下）、grant_type授权类型去请求路径<https://api.weixin.qq.com/sns/jscode2session>，来获取session_key。
- 3 拿到session_key可以生成自己的3rd_session存储在storage。
- 4 后续用户进入微信小程序，先从storage获得3rd_session，再根据这个去查找合法的session_key。

登录时序图如图4.28所示。

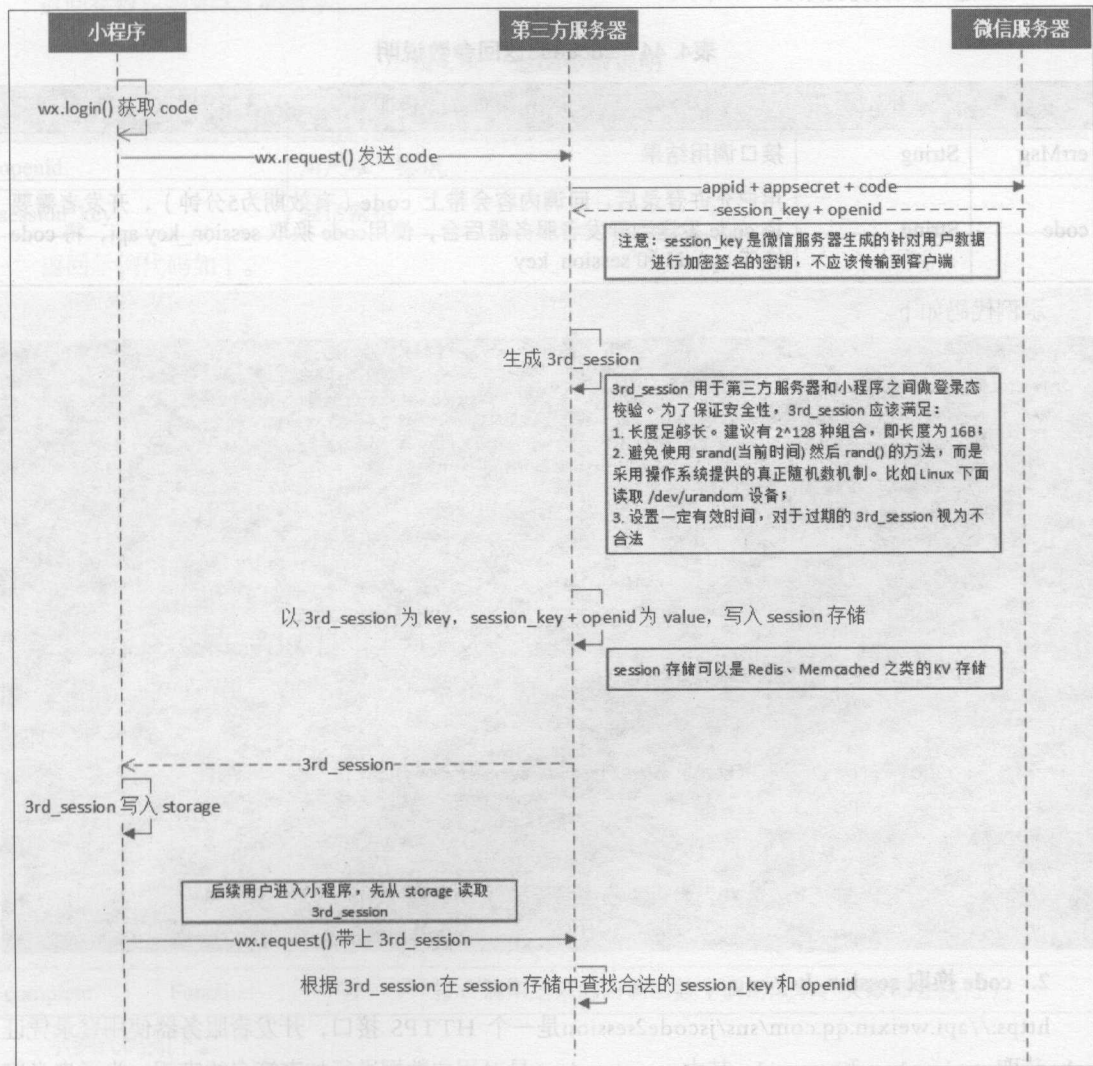


图4.28 登录时序图

1. wx.login(OBJECT)获取登录凭证code

微信小程序使用wx.login接口来获取登录凭证（code），进而换取用户登录态信息，包括用户的唯一标识（openid）及本次登录的会话密钥（session_key）。用户数据的加解密通信需要依赖会话密钥完成。具体参数说明如表4.43所示。

表4.43 wx.login参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.44所示。

表4.44 success返回参数说明

| 参数 | 类型 | 说明 |
|--------|--------|--|
| errMsg | String | 接口调用结果 |
| code | String | 用户允许登录后，回调内容会带上 code（有效期为5分钟），开发者需要将 code 发送到开发者服务器后台，使用code 换取 session_key api，将 code 换成 openid 和 session_key |

示例代码如下。

```
App({
  onLaunch: function() {
    wx.login({
      success: function(res) {
        if (res.code) {
          // 发起网络请求
          wx.request({
            url: 'https://test.com/onLogin',
            data: {
              code: res.code
            }
          })
        } else {
          console.log('获取用户登录态失败！' + res.errMsg)
        }
      }
    });
  }
});
```

2. code 换取 session_key

https://api.weixin.qq.com/sns/jscode2session是一个 HTTPS 接口，开发者服务器使用登录凭证code 获取 session_key 和 openid。其中，session_key 是对用户数据进行加密签名的密钥。为了自身应用安全，session_key 不应该在网络上传输。

接口地址为`https://api.weixin.qq.com/sns/jscode2session?Appid=APPID&secret=SECRET&js_code=JSCODE&grant_type=authorization_code`。

参数说明如表4.45所示。

表4.45 接口参数说明

| 属性 | 必填 | 说明 |
|------------|----|-------------------------------------|
| Appid | 是 | 小程序唯一标识 |
| secret | 是 | 小程序的 App secret |
| js_code | 是 | 登录时获取的 code |
| grant_type | 是 | 填写为 <code>authorization_code</code> |

返回参数说明如表4.46所示。

表4.46 返回参数说明

| 参数 | 说明 |
|-------------|--------|
| openid | 用户唯一标识 |
| session_key | 会话密钥 |

返回示例代码如下。

```
// 正常返回的 JSON 数据包
{
  "openid": "OPENID",
  "session_key": "SESSIONKEY"
}
// 错误时返回 JSON 数据包（示例为 code 无效）
{
  "errcode": 40029,
  "errmsg": "invalid code"
}
```

3. wx.checkSession(OBJECT) 检查登录状态是否过期

微信小程序可以使用`wx.checkSession(OBJECT)` 检查登录状态是否过期，如果过期就重新登录，具体参数说明如表4.47所示。

表4.47 wx.checkSession参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| success | Function | 否 | 接口调用成功的回调函数，登录态未过期 |
| fail | Function | 否 | 接口调用失败的回调函数，登录态已过期 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
wx.checkSession({
  success: function(){
    // 登录状态未过期
```

```
},
fail: function(){
    // 登录状态过期
    wx.login()
}
})
```

4. wx.getUserInfo(OBJECT) 获取用户信息

微信小程序使用wx.getUserInfo(OBJECT)来获取用户信息，在获取用户信息之前，需要调用wx.login 接口，只有用户在登录状态，才能获取到用户的相关信息。具体参数说明如表4.48所示。

表4. 48 wx. getUserInfo参数说明

| 属性 | 类型 | 必填 | 说明 |
|----------|----------|----|--------------------------|
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

success返回参数说明如表4.49所示。

表4. 49 success返回参数说明

| 属性 | 类型 | 说明 |
|---------------|--------|--|
| userInfo | OBJECT | 用户信息对象，不包含 openid 等敏感信息 |
| rawData | String | 不包括敏感信息的原始数据字符串，用于计算签名 |
| signature | String | 使用 sha1(rawData + sessionkey) 得到字符串，用于校验用户信息 |
| encryptedData | String | 包括敏感数据在内的完整用户信息的加密数据，详见加密数据解密算法 |
| iv | String | 加密算法的初始向量，详见加密数据解密算法 |

示例代码如下。

```
Page({
  onLoad: function () {
    wx.getUserInfo({
      success: function (res) {
        console.log(res);
        var userInfo = res.userInfo
        var nickName = userInfo.nickName
        var avatarUrl = userInfo.avatarUrl
        var gender = userInfo.gender // 性别 0：未知、1：男、2：女
        var province = userInfo.province
        var city = userInfo.city
        var country = userInfo.country
      }
    })
  }
})
```


4.11 微信支付API

微信支付主要有5个步骤：小程序内调用登录接口、商户server调用支付统一下单、商户server调用再次签名、商户server接收支付通知、商户server查询支付结果，小程序支付的交互过程如图4.29所示。

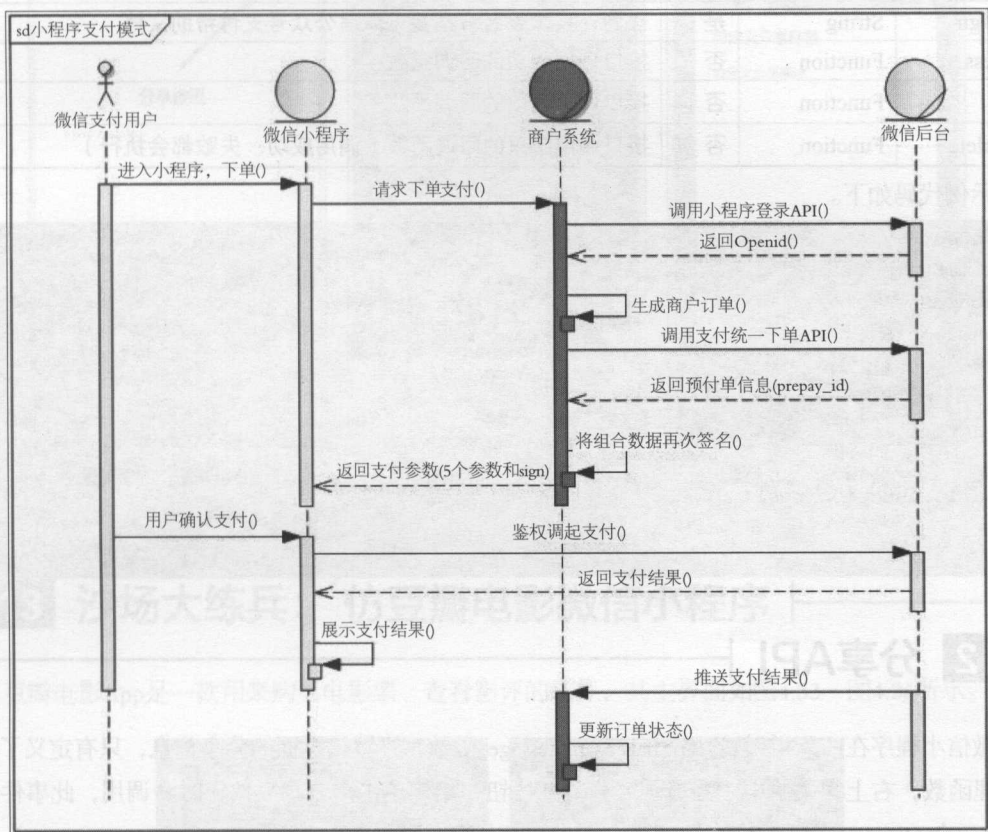


图4.29 小程序支付交互时序图

说明：

1 商户server调用支付统一下单，API参见公共API：https://pay.weixin.qq.com/wiki/doc/api/wxa/wxa_api.php?chapter=9_1。

2 商户server调用再次签名，API参见公共API：https://pay.weixin.qq.com/wiki/doc/api/wxa/wxa_api.php?chapter=7_7。

3 商户server接收支付通知，API参见公共API：https://pay.weixin.qq.com/wiki/doc/api/wxa/wxa_api.php?chapter=9_7。

4 商户server查询支付结果，API参见公共API：https://pay.weixin.qq.com/wiki/doc/api/wxa/wxa_api.php?chapter=9_2。

微信小程序提供了微信支付接口，可以使用wx.requestPayment(OBJECT)来进行微信支付，具体参数说明如表4.50所示。

表4. 50 wx.requestPayment小程序支付参数说明

| 属性 | 类型 | 必填 | 说明 |
|-----------|----------|----|--|
| timeStamp | String | 是 | 时间戳从1970年1月1日00:00:00至今的秒数，即当前时间 |
| nonceStr | String | 是 | 随机字符串，长度为32个字符以下 |
| package | String | 是 | 统一下单接口返回的 prepay_id 参数值，提交格式如prepay_id=* |
| signType | String | 是 | 签名算法，暂支持 MD5 |
| paySign | String | 是 | 签名，具体签名方案参见微信公众号支付帮助文档 |
| success | Function | 否 | 接口调用成功的回调函数 |
| fail | Function | 否 | 接口调用失败的回调函数 |
| complete | Function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |

示例代码如下。

```
wx.requestPayment({
  'timeStamp': '',
  'nonceStr': '',
  'package': '',
  'signType': 'MD5',
  'paySign': '',
  'success':function(res){
  },
  'fail':function(res){
  }
})
```

4.12 分享API

微信小程序在Page 中定义 onShareAppMessage 函数，设置该页面的分享信息，只有定义了此事件处理函数，右上角菜单中才会显示“分享”按钮，用户单击“分享”按钮时会调用，此事件需要return 一个 Object，用于自定义分享内容，自定义分享字段如表4.51所示。

表4. 51 自定义分享字段

| 字段 | 说明 | 默认值 |
|-------|------|-----------------------------|
| title | 分享标题 | 当前小程序名称 |
| desc | 分享描述 | 当前小程序名称 |
| path | 分享路径 | 当前页面 path ，必须是以 "/" 开头的完整路径 |

示例代码如下。

```
Page({
  onShareAppMessage: function () {
    return {
      title: '自定义分享标题',
      desc: '自定义分享描述',
      path: '/page/user?id=123'
    }
  }
})
```

界面效果如图4.30~图4.32所示。

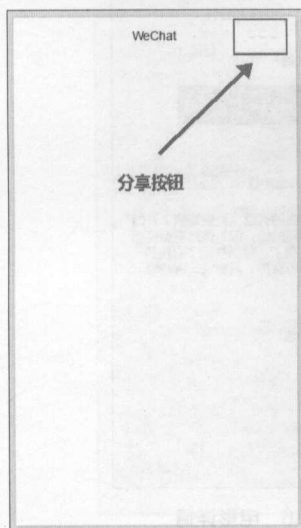


图4.30 分享按钮

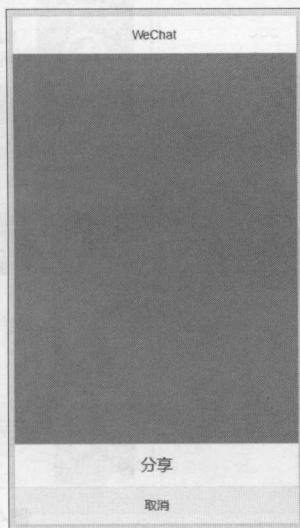


图4.31 是否分享

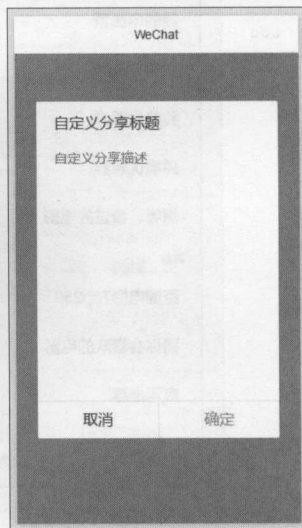


图4.32 分享界面

4.13 沙场大练兵：仿豆瓣电影微信小程序

豆瓣电影App是一款用来购买电影票、查看影评的软件，其主界面如图4.33~图4.36所示。



图4.33 电影

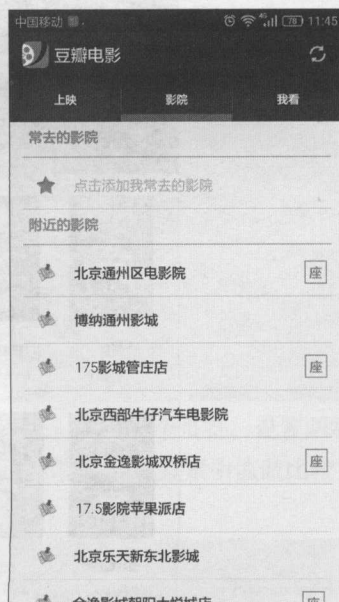


图4.34 影院

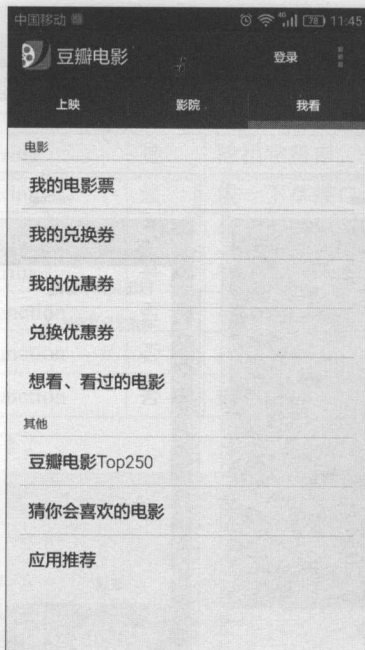


图4.35 我看



图4.36 电影详情

下面，我们来设计一款豆瓣电影微信小程序，以查看上映的电影以及电影详情。

4.13.1 电影顶部页签切换效果

在电影界面的顶部有3个页签：上映、影院、我看，页签的切换，会带动相应的内容进行切换展示。我们采用顶部页签切换效果，来完成各个页面的切换展示，如图4.37所示。



图4.37 顶部页签

1 添加一个AppID为wxa7730e0596be9404的douban项目，如图4.38所示。

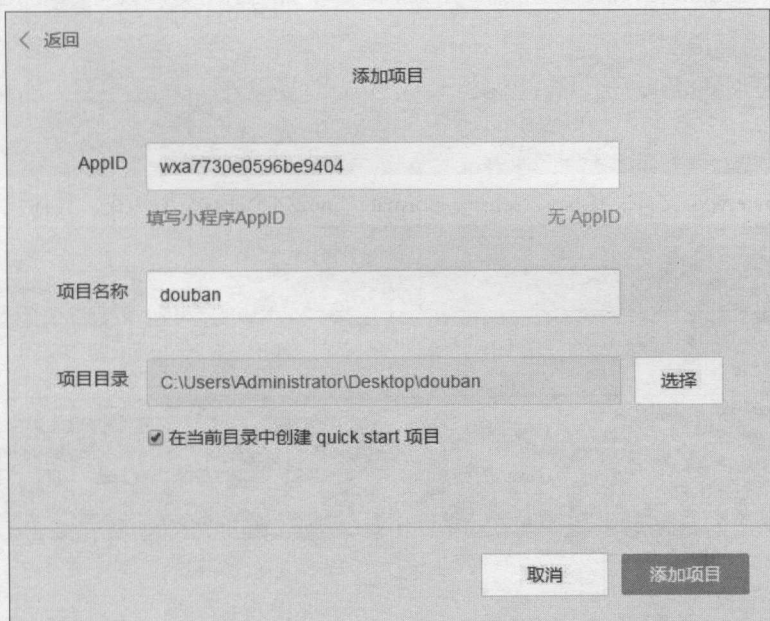


图4.38 添加douban项目

2 进入App.json文件，添加“pages/movie/movie”“pages/cinema/cinema”“pages/me/me”“pages/movieDetail/movieDetail”4个文件路径，删除默认创建的index、logs文件路径，将窗口导航栏的背景色设置为黑色（#1A1A1A），导航标题为“豆瓣电影”，文字颜色设置为白色（white），具体代码如下。

```
{
  "pages": [
    "pages/movie/movie",
    "pages/cinema/cinema",
    "pages/me/me",
    "pages/movieDetail/movieDetail"
  ],
  "window": {
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#1A1A1A",
    "navigationBarTitleText": "豆瓣电影",
    "navigationBarTextStyle": "white"
  }
}
```

3 进入movie.wxml文件，采用view视图容器布局顶部的3个页签，设置两种样式，一种是选中样式select，另一种是正常样式normal，定义currentTab变量与页签索引值做比较，id为索引值，添加switchNav绑定事件，具体代码如下。

```
<view class="nav">
  <view id="0" class="{currentTab == 0?'select':'normal'}" bindtap="switchNav">上映
</view>
```

```
<view class="line">|</view>
<view id="1" class="{{currentTab == 1?'select':'normal'}}" bindtap="switchNav">影院
</view>
<view class="line">|</view>
<view id="2" class="{{currentTab == 2?'select':'normal'}}" bindtap="switchNav">我看
</view>
</view>
```

4 进入movie.wxss文件，给nav、select、normal、line这4个class添加样式，具体代码如下。

```
.nav{
  display: flex;
  flex-direction: row;
  background-color: #222222;
}
.select{
  width: 32%;
  height: 45px;
  line-height: 45px;
  text-align: center;
  color: #ffffff;
  font-size: 13px;
  border-bottom: 10px solid #777777;
}
.normal{
  width: 32%;
  height: 45px;
  line-height: 45px;
  text-align: center;
  color: #ffffff;
  font-size: 13px;
}
.line{
  height: 45px;
  line-height: 45px;
  font-size: 25px;
  color: #666666;
}
```

界面效果如图4.39所示。

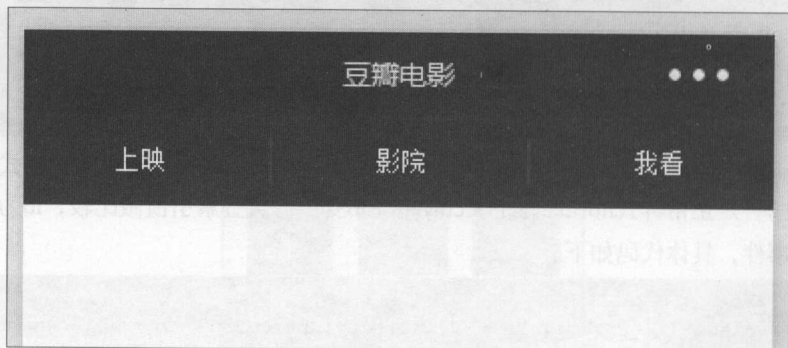


图4.39 顶部页签布局效果

5 进入movie.js文件，定义currentTab变量默认值为0，添加switchNav页签单击事件，单击时获取页签的id索引值，然后赋值给currentTab变量，具体代码如下。

```
Page({
  data: {
    currentTab: 0
  },
  switchNav: function (e) {
    var id = e.currentTarget.id;
    this.setData({ currentTab: id });
  }
})
```

这样就可以对顶部页签进行相应的切换显示，如图4.40和图4.41所示。

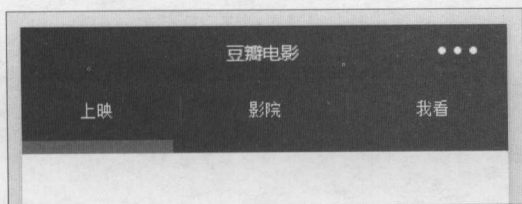


图4.40 “上映”页签选中

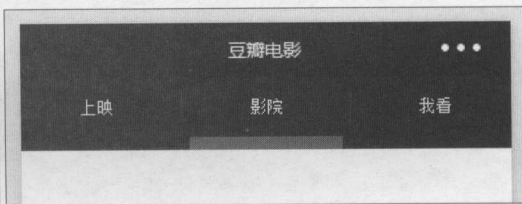


图4.41 “影院”页签选中

6 进入movie.wxml文件，布局页签内容，单击页签切换时，页签内容滑动进行切换，需要借助于swiper滑块视图组件，根据currentTab变量的索引值来判断显示swiper哪个面板内容，具体代码如下。

```
<view class="nav">
  <view id="0" class="{{currentTab == 0?'select':'normal'}}" bindtap="switchNav"> 上映
</view>
  <view class="line">|</view>
  <view id="1" class="{{currentTab == 1?'select':'normal'}}" bindtap="switchNav"> 影院
</view>
  <view class="line">|</view>
  <view id="2" class="{{currentTab == 2?'select':'normal'}}" bindtap="switchNav"> 我看
</view>
</view>
<swiper current="{{currentTab}}" style="height:{{winHeight}}px">
  <swiper-item>
    <view> 上映内容 </view>
  </swiper-item>
  <swiper-item>
    <view> 影院内容 </view>
  </swiper-item>
  <swiper-item>
    <view> 我看内容 </view>
  </swiper-item>
</swiper>
```

7 进入movie.js文件，添加onLoad生命周期函数，使用wx.getSystemInfo获得窗口的高度和宽度，赋值给变量winWidth、winHeight，winHeight作为页签内容的高度，具体代码如下。

```

Page({
  data: {
    currentTab: 0,
    winWidth: 0,
    winHeight: 0
  },
  onLoad: function (e) {
    var page = this;
    wx.getSystemInfo({
      success: function (res) {
        console.log(res);
        page.setData({ winWidth: res.windowWidth });
        page.setData({ winHeight: res.windowHeight });
      }
    })
  },
  switchNav: function (e) {
    var id = e.currentTarget.id;
    this.setData({ currentTab: id });
  }
})

```

这样就实现了顶部页签的切换效果，页签切换时，页签内容也连带一起切换，实现动态切换效果的展示。

4.13.2 电影海报轮播效果

海报轮播效果是很多App软件和网站都会采用的一种方式，在有限的区域内动态地展示商品图片信息或者广告信息，豆瓣电影里也有海报轮播效果，如图4.42所示。



图4.42 海报轮播区域

1 将海报轮播的图片拷贝到douban项目，进入到movie.wxml文件里，进行海报轮播效果的界面布局，具体代码如下。

```
<view class="nav">
  <view id="0" class="{{currentTab == 0?'select':'normal'}}" bindtap="switchNav">上映
</view>
  <view class="line">|</view>
  <view id="1" class="{{currentTab == 1?'select':'normal'}}" bindtap="switchNav">影院
</view>
  <view class="line">|</view>
  <view id="2" class="{{currentTab == 2?'select':'normal'}}" bindtap="switchNav">我看
</view>
</view>
<swiper current="{{currentTab}}" style="height:{{winHeight}}px">
  <swiper-item>
    <view class="haibao">
      <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval=
        "{{interval}}" duration="{{duration}}" style="height:74px;">
        <block wx:for="{{imgUrls}}">
          <swiper-item>
            <image src="{{item}}" class="silde-image" style="width:100%;height:74px;">
</image>
          </swiper-item>
        </block>
      </swiper>
    </view>
  </swiper-item>
  <swiper-item>
    <view>影院内容</view>
  </swiper-item>
  <swiper-item>
    <view>我看内容</view>
  </swiper-item>
</swiper>
```

2 进入movie.js文件，定义海报轮播需要的变量值indicatorDots为false、autoplay为true、interval为5000、duration为1000、imgUrls为“/images/haibao/1.jpg”“/images/haibao/2.jpg”“/images/haibao/3.jpg”，具体代码如下。

```
Page({
  data: {
    currentTab: 0,
    winWidth: 0,
    winHeight: 0,
    indicatorDots: false,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [
```



```
    "/images/haibao/1.jpg",  
    "/images/haibao/2.jpg",  
    "/images/haibao/3.jpg",  
    "/images/haibao/4.jpg"  
  ],  
  },  
  onLoad: function (e) {  
    var page = this;  
    wx.getSystemInfo({  
      success: function (res) {  
        console.log(res);  
        page.setData({ winWidth: res.windowWidth });  
        page.setData({ winHeight: res.windowHeight });  
      }  
    })  
  },  
  switchNav: function (e) {  
    var id = e.currentTarget.id;  
    this.setData({ currentTab: id });  
  }  
})
```

这样就实现了海报轮播效果，界面效果如图4.43所示。

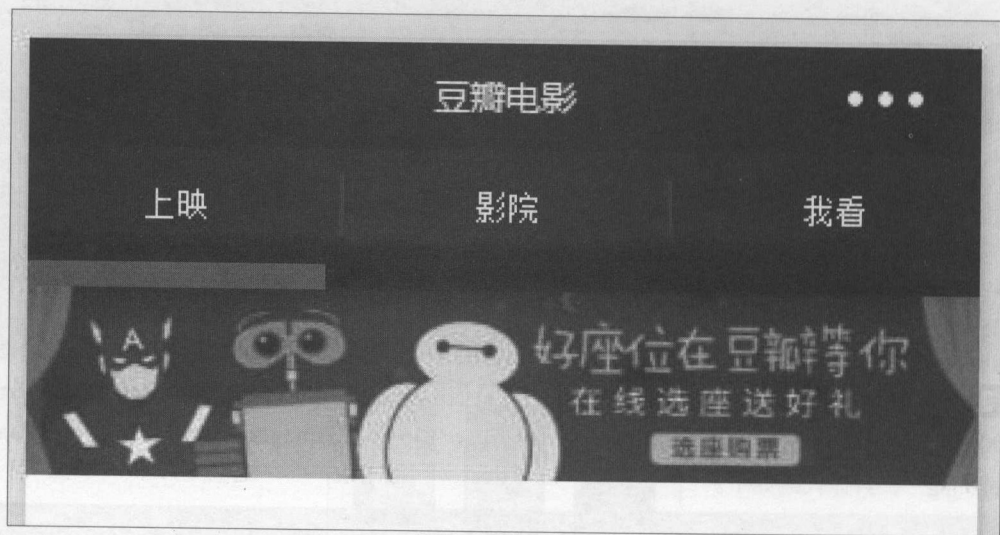


图4.43 海报轮播效果

4.13.3 电影列表方式布局

豆瓣电影的电影列表采用每行3列的方式来进行布局，显示电影海报和电影名称，如图4.44所示。



图4.44 电影列表

1 进入movie.js文件，定义loadMovies函数使用wx.request来获取到豆瓣电影的电影列表信息，在onLoad函数里调用loadMovies函数，定义movies变量，将电影列表赋值给movies，具体代码如下。

```
Page({
  data: {
    currentTab: 0,
    winWidth: 0,
    winHeight: 0,
    indicatorDots: false,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [
      "/images/haibao/1.jpg",
      "/images/haibao/2.jpg",
      "/images/haibao/3.jpg",
      "/images/haibao/4.jpg"
    ],
  },
  movies: [],
},
onLoad: function (e) {
  var page = this;
  wx.getSystemInfo({
```

```

    success: function (res) {
      console.log(res);
      page.setData({ winWidth: res.windowWidth });
      page.setData({ winHeight: res.windowHeight });
    }
  });
  this.loadMovies();
},
switchNav: function (e) {
  var id = e.currentTarget.id;
  this.setData({ currentTab: id });
},
loadMovies: function () {
  var page = this;
  wx.request({
    url: 'https://api.douban.com/v2/movie/in_theaters',
    method: 'GET',
    header: {
      "Content-Type": "json"
    },
    success: function (res) {
      var subjects = res.data.subjects;
      var size = subjects.length; // 电影总数量
      var len = parseInt(size / 3); // 每行放置 3 个电影，计算出需要多少行

      console.log(len);
      console.log(subjects);
      page.setData({ movies: subjects });
      page.setData({ winHeight: (len + 1) * 230 }); // 动态地设置电影内容的高度
    }
  })
}
})

```

2 获取到电影列表信息后，进入movie.wxml文件，电影海报图片采用中等大小的图片，宽度设置为100，高度设置为150，绑定loadDetail查看电影详情，具体代码如下。

```

<view class="nav">
  <view id="0" class="{{currentTab == 0?'select':'normal'}}" bindtap="switchNav">上映
</view>
  <view class="line">|</view>
  <view id="1" class="{{currentTab == 1?'select':'normal'}}" bindtap="switchNav">影院
</view>
  <view class="line">|</view>
  <view id="2" class="{{currentTab == 2?'select':'normal'}}" bindtap="switchNav">我看
</view>
</view>
<swiper current="{{currentTab}}" style="height:{{winHeight}}px">
  <swiper-item>
    <view class="haibao">

```



```

<swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval=
"{{interval}}" duration="{{duration}}" style="height:74px;">
  <block wx:for="{{imgUrls}}">
    <swiper-item>
      <image src="{{item}}" class="slide-image" style="width:100%;height:74px;">
</image>
    </swiper-item>
  </block>
</swiper>
</view>
<view class="items">
  <block wx:for="{{movies}}">
    <view class="item" bindtap="loadDetail" id="{{item.id}}">
      <view>
        <image src="{{item.images.medium}}" style="width:100px;height:150px;"></
image>
      </view>
      <view class="name">
        <text>{{item.title}}</text>
      </view>
    </view>
  </block>
</view>
</swiper-item>
<swiper-item>
  <view>影院内容</view>
</swiper-item>
<swiper-item>
  <view>我看内容</view>
</swiper-item>
</swiper>

```

3 进入movie.wxss文件，给items、item、name这3个class添加样式，具体代码如下。

```

.nav{
  display: flex;
  flex-direction: row;
  background-color: #222222;
}
.select{
  width: 32%;
  height: 45px;
  line-height: 45px;
  text-align: center;
  color: #ffffff;
  font-size: 13px;
  border-bottom: 10px solid #777777;
}
.normal{
  width: 32%;
  height: 45px;
  line-height: 45px;

```

```

text-align: center;
color: #ffffff;
font-size: 13px;
}
.line{
height: 45px;
line-height: 45px;
font-size: 25px;
color: #666666;
}
.items{
background-color: #f2f2f2;
height: 1000px;
}
.item{
width: 32%;
height: 200px;
margin-top: 10px;
text-align: center;
float: left;
}
.name{
font-size: 14px;
font-weight: bold;
margin: 5px;
}

```

这样就完成了电影列表界面的布局展示，界面效果如图4.45所示。



图4.45 电影列表布局

4.13.4 电影详情页布局

在电影列表界面里，单击电影海报图片，可以查看具体的电影详情。电影详情页在顶部也是采用页签切换的方式进行布局，布局方式和电影页面一致。页签的下面是介绍电影相关信息的区域，接着是“我想看”和“看过了”两个按钮，再往下是电影介绍、导演演员列表的展现，如图4.46所示。



图4.46 电影详情页

1 进入movie.js文件，添加loadDetail函数，跳转到详情页里需要把电影的id带过去，详情信息需要电影id才能查得到，具体代码如下。

```
Page({
  data: {
    currentTab: 0,
    winWidth: 0,
    winHeight: 0,
    indicatorDots: false,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [
      "/images/haibao/1.jpg",
      "/images/haibao/2.jpg",
```



```
    "/images/haibao/3.jpg",
    "/images/haibao/4.jpg"
  ],
  onLoad: function (e) {
    var page = this;
    wx.getSystemInfo({
      success: function (res) {
        console.log(res);
        page.setData({ winWidth: res.windowWidth });
        page.setData({ winHeight: res.windowHeight });
      }
    });
    this.loadMovies();
  },
  switchNav: function (e) {
    var id = e.currentTarget.id;
    this.setData({ currentTab: id });
  },
  loadMovies: function () {
    var page = this;
    wx.request({
      url: 'https://api.douban.com/v2/movie/in_theaters',
      method: 'GET',
      header: {
        "Content-Type": "json"
      },
      success: function (res) {
        var subjects = res.data.subjects;
        var size = subjects.length; // 电影总数量
        var len = parseInt(size / 3); // 每行放置 3 个电影，计算出需要多少行

        console.log(len);
        console.log(subjects);
        page.setData({ movies: subjects });
        page.setData({ winHeight: (len + 1) * 230 }); // 动态地设置电影内容的高度
      }
    })
  },
  loadDetail: function (e) {
    var id = e.currentTarget.id;
    wx.navigateTo({
      url: '../movieDetail/movieDetail?id=' + id
    })
  }
})
```

2 进入movieDetail.js文件，定义3个变量：电影movie、导演directors、演员casts。在onLoad函数中，根据传递过来的id查找电影详情，然后赋值给movie、directors、casts，具体代码如下。

```

Page({
  data: {
    movie: {},
    directors: [],
    casts: []
  },
  onLoad: function(e) {
    var page = this;
    wx.request({
      url: 'https://api.douban.com/v2/movie/subject/' + e.id,
      header: {
        "Content-Type": "json"
      },
      success: function(res) {
        console.log(res);
        var movie = res.data;
        page.setData({movie: movie});
        page.setData({directors: movie.directors});
        page.setData({casts: movie.casts});
        wx.setNavigationBarTitle({
          title: movie.title
        })
      }
    })
  }
})

```

3 进入movieDetail.wxml文件，进行顶部页签界面的布局以及页签内容的布局，具体代码如下。

```

<view class="nav">
  <view id="0" class="{{currentTab == 0?'select':'normal'}}" bindtap="switchNav"> 介绍
</view>
  <view class="line">|</view>
  <view id="1" class="{{currentTab == 1?'select':'normal'}}" bindtap="switchNav"> 图片
</view>
  <view class="line">|</view>
  <view id="2" class="{{currentTab == 2?'select':'normal'}}" bindtap="switchNav"> 短评
</view>
  <view class="line">|</view>
  <view id="3" class="{{currentTab == 3?'select':'normal'}}" bindtap="switchNav"> 影评
</view>
</view>
<swiper current="{{currentTab}}" style="height:1200px;background-color:#F9F9F9;">
  <swiper-item>
    <view> 介绍内容 </view>
  </swiper-item>
  <swiper-item>
    <view> 图片内容 </view>
  </swiper-item>
  <swiper-item>

```



```
<view> 短评内容 </view>
</swiper-item>
<swiper-item>
  <view> 影评内容 </view>
</swiper-item>
</swiper>
```

4 进入movieDetail.wxss文件，给class添加相应的样式，具体代码如下。

```
.nav{
  display: flex;
  flex-direction: row;
  background-color: #222222;
}
.select{
  width: 32%;
  height: 30px;
  line-height: 30px;
  text-align: center;
  color: #ffffff;
  font-size: 13px;
  border-bottom: 8px solid #777777;
}
.normal{
  width: 32%;
  height: 30px;
  line-height: 30px;
  text-align: center;
  color: #ffffff;
  font-size: 13px;
}
.line{
  height: 30px;
  line-height: 30px;
  font-size: 15px;
  color: #666666;
}
```

5 进入movieDetail.js文件，定义变量currentTab、winWidth、winHeight，并进行赋值，具体代码如下。

```
Page({
  data: {
    currentTab: 0,
    winWidth: 0,
    winHeight: 0,
    movie: {},
    directors: [],
    casts: []
  },
  onLoad: function (e) {
    var page = this;
```



```

wx.request({
  url: 'https://api.douban.com/v2/movie/subject/' + e.id,
  header: {
    "Content-Type": "json"
  },
  success: function (res) {
    console.log(res);
    var movie = res.data;
    page.setData({ movie: movie });
    page.setData({ directors: movie.directors });
    page.setData({ casts: movie.casts });
    wx.setNavigationBarTitle({
      title: movie.title
    })
  }
});
wx.getSystemInfo({
  success: function (res) {
    console.log(res);
    page.setData({ winWidth: res.windowWidth });
    page.setData({ winHeight: res.windowHeight });
  }
});
},
switchNav: function (e) {
  var id = e.currentTarget.id;
  this.setData({ currentTab: id });
}
})

```

界面效果如图4.47所示。

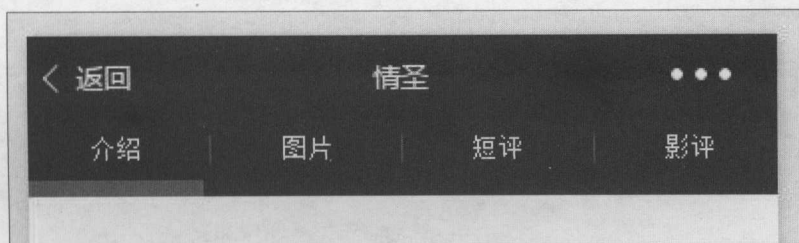


图4.47 顶部页签切换效果

6 进入movieDetail.wxml文件，进行电影详情、电影简介、导演和演员的界面布局设计，具体代码如下。

```

<view class="nav">
  <view id="0" class="{{currentTab == 0?'select':'normal'}}" bindtap="switchNav"> 介绍
</view>
  <view class="line"></view>
  <view id="1" class="{{currentTab == 1?'select':'normal'}}" bindtap="switchNav"> 图片
</view>
  <view class="line"></view>

```

```

<view id="2" class="{{currentTab == 2?'select':'normal'}}" bindtap="switchNav"> 短评
</view>
<view class="line">|</view>
<view id="3" class="{{currentTab == 3?'select':'normal'}}" bindtap="switchNav"> 影评
</view>
</view>
<swiper current="{{currentTab}}" style="height:1200px;background-color:#F9F9F9;">
  <swiper-item>
    <view class="movieInfo">
      <view>
        <image src="{{movie.images.medium}}" style="width:100px;height:150px"></image>
      </view>
      <view class="detail">
        <view>
          <text class="score">评分: {{movie.rating.average}}</text> ({{movie.ratings_
count}} 人评价) </view>
          <view>
            <text>{{movie.year}} 年上映 </text>
          </view>
          <view>
            <text class="desc">{{movie.genres[0]}}</text>
          </view>
          <view>
            <text class="desc">{{movie.countries[0]}}</text>
          </view>
          <view class="buy">选座购票 </view>
        </view>
      </view>
      <view class="opr">
        <view>我想看 </view>
        <view>看过了 </view>
      </view>
      <view class="intro">
        <text>{{movie.summary}}</text>
      </view>
      <block wx:for="{{directors}}">
        <view class="personInfo">
          <view>
            <image src="{{item.avatars.small}}" style="width:70px;height:100px"></image>
          </view>
          <view class="name">
            <view>
              <text>{{item.name}} [ 导演 ]</text>
            </view>
          </view>
        </view>
      </block>
      <block wx:for="{{casts}}">
        <view class="personInfo">
          <view>

```

```

        <image src="{{item.avatars.small}}" style="width:70px;height:100px"></image>
    </view>
    <view class="name">
        <view>
            <text>{{item.name}}</text>
        </view>
    </view>
</view>
</block>
</swiper-item>
<swiper-item>
    <view> 图片内容 </view>
</swiper-item>
<swiper-item>
    <view> 短评内容 </view>
</swiper-item>
<swiper-item>
    <view> 影评内容 </view>
</swiper-item>
</swiper>

```

7 进入movieDetail.wxss文件，给class添加相应的样式，具体代码如下。

```

.nav{
    display: flex;
    flex-direction: row;
    background-color: #222222;
}
.select{
    width: 32%;
    height: 30px;
    line-height: 30px;
    text-align: center;
    color: #ffffff;
    font-size: 13px;
    border-bottom: 8px solid #777777;
}
.normal{
    width: 32%;
    height: 30px;
    line-height: 30px;
    text-align: center;
    color: #ffffff;
    font-size: 13px;
}
.line{
    height: 30px;
    line-height: 30px;
    font-size: 15px;
    color: #666666;
}
.movieInfo{

```



```

    display: flex;
    flex-direction: row;
    margin: 15px;
}
.detail{
    margin-left: 15px;
    font-size: 13px;
}
.detail view{
    margin-bottom: 7px;
}
.score{
    color: #FC7F60;
}
.desc{
    font-weight: bold;
}
.buy{
    width: 155px;
    height: 40px;
    line-height: 40px;
    background-color: #2DADDC;
    text-align: center;
    color: #ffffff;
}
.opr{
    display: flex;
    flex-direction: row;
}
.opr view{
    width: 45%;
    height: 40px;
    line-height: 40px;
    margin: 0 auto;
    background-color: #EEEEEE;
    font-size: 13px;
    text-align: center;
    border-radius: 3px;
}
.intro{
    background-color: #F1F1F1;
    margin: 10px;
    font-size: 13px;
    padding: 10px;
    line-height: 15px;
}
.personInfo{
    display: flex;
    flex-direction: row;
    margin: 15px;

```

```
background-color: #F1F1F1;
}
.name{
  font-size: 14px;
  font-weight: bold;
  line-height: 100px;
  margin-left: 10px;
}
```

这样就实现了电影详情页界面的设计，界面效果如图4.48所示。



图4.48 电影详情页界面

豆瓣电影微信小程序主要实现了页签的切换效果、海报轮播效果、豆瓣电影列表的展现、电影详情页的设计，需要用到view视图内容组件、image图片组件、swiper滑块视图组件等，同时通过wx.request、wx.getSystemInfo等API接口，进行网络通信获取、分析json数据，并将其展现在页面里。

4.13.5 项目上传与预览

项目开发完后，可以上传到微信小程序服务器上，如图4.49所示。



图4.49 项目上传

可以在手机上预览豆瓣电影微信小程序，如图4.50所示。



图4.50 项目预览

需要扫描二维码，才能在手机上进行预览，预览效果如图4.51和图5.52所示。



图4.51 电影列表

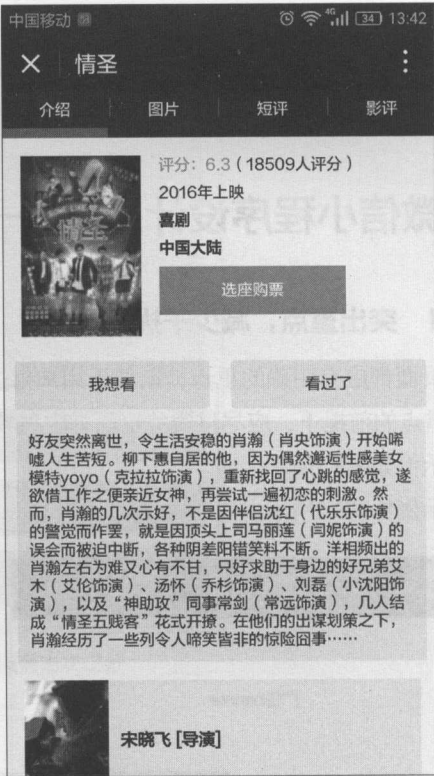


图4.52 电影详情

4.14 小结

本章主要学习微信小程序API的使用，重点应该掌握以下内容。

- 1 掌握微信小程序如何请求服务器数据。
- 2 掌握微信小程序文件上传、下载和WebSocket会话的使用。
- 3 掌握微信小程序图片处理、文件操作、数据缓存的API。
- 4 了解微信小程序位置信息、设备应用API的使用。
- 5 掌握微信小程序交互反馈、登录、微信支付、分享API的使用。

第5章 微信小程序设计及问答

5.1 微信小程序设计

5.1.1 突出重点，减少干扰项

每个页面都应有明确的重点，以便于用户每进入一个新页面的时候都能快速地理解页面内容。在确定了重点的前提下，应尽量避免页面上出现其他与用户的决策和操作无关的干扰因素。

反面示例如图5.1所示。

正确示例如图5.2所示。

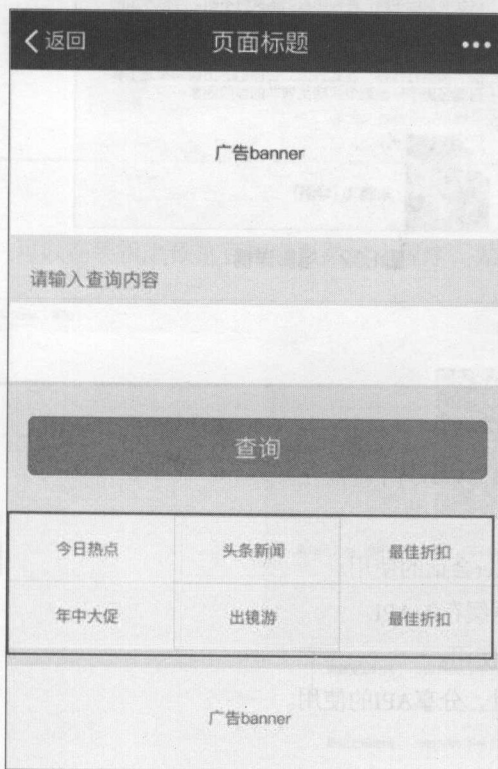


图5.1 干扰项过多

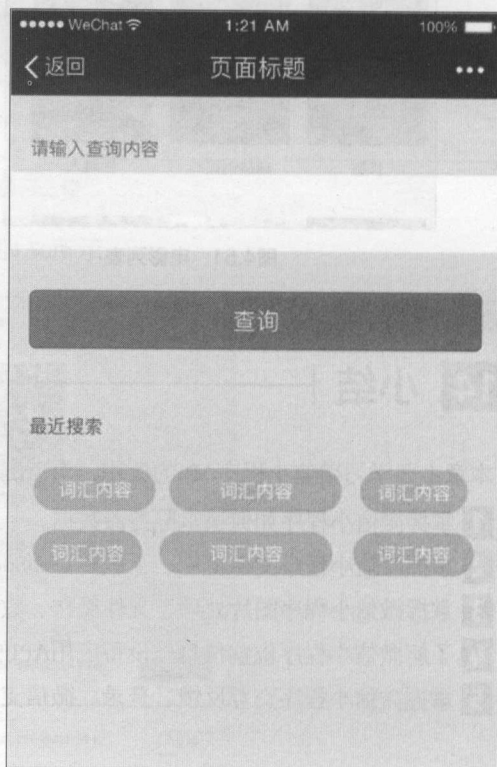


图5.2 减少干扰项

5.1.2 主次动作区分明显

在一个界面上有多个按钮的时候，按钮设计有主次之分，并且区分明显，让用户看到后就知道他应该做什么、该怎么做。

反面示例如图5.3所示。

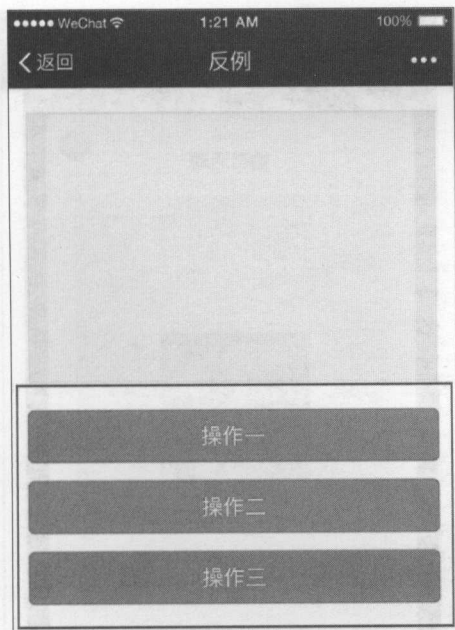


图5.3 按钮没有主次之分

正确示例如图5.4所示。

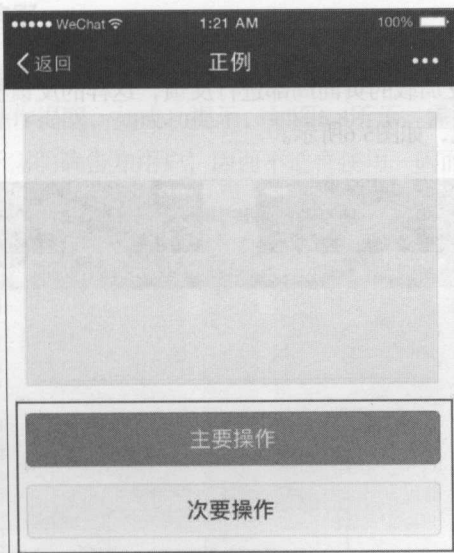


图5.4 按钮有主次之分

5.1.3 流程明确，避免打断

用户在进入某个页面进行某一个操作流程时，应避免出现用户目标流程之外的内容而打断用户。

例如，用户想进入某个页面进行商品购买，突然弹出抽奖的模态窗口界面，等用户抽完奖，也许就忘记了要去买商品这件事，显然，这对我们引导用户购买商品很不利，所有应尽量避免打断用户的主要流程操作，如图5.5所示就是抽奖打断用户操作。

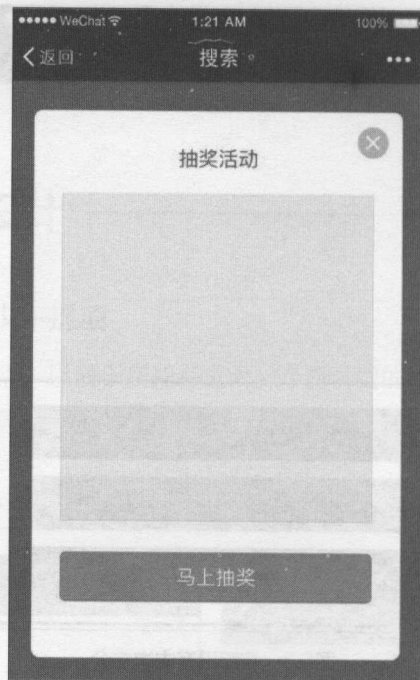


图5.5 抽奖打断用户操作

5.1.4 局部加载反馈

局部加载反馈即只在触发加载的页面局部进行反馈，这样的反馈机制更加有针对性，且页面跳动小，是微信推荐的反馈方式，如图5.6所示。

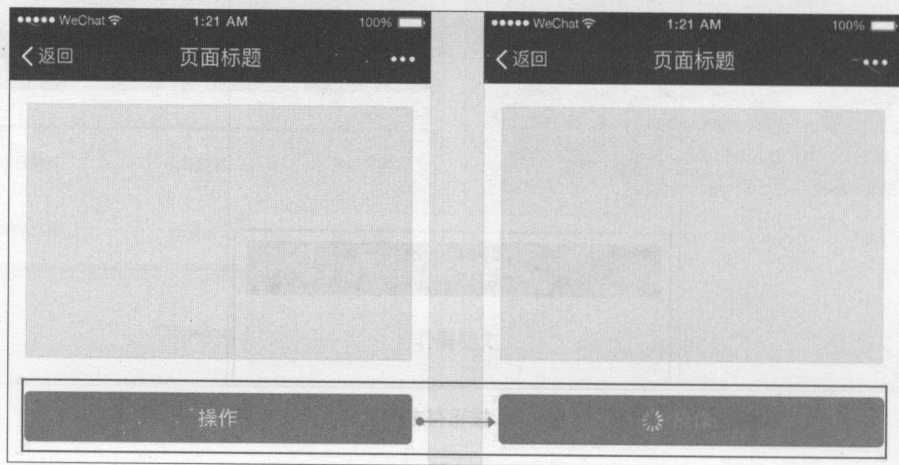


图5.6 局部加载反馈

5.1.5 模态窗口加载反馈

模态的加载样式将覆盖整个页面，由于无法明确告知具体加载位置或内容，所以可能会引起用户的焦虑感，因此应谨慎使用，如图5.7所示。除了在某些全局性操作下之外，不要使用模态的加载。

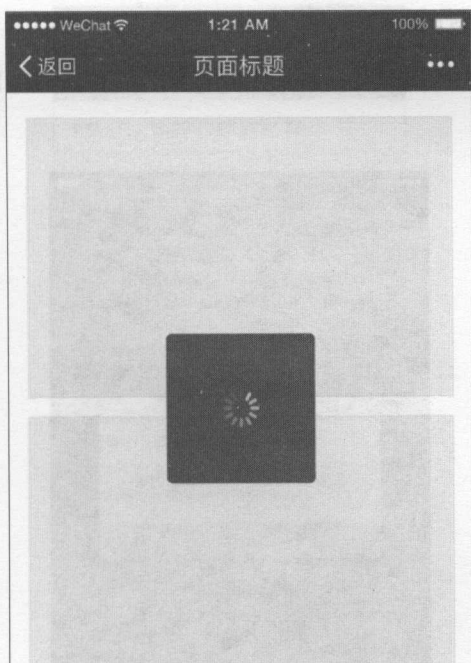


图5.7 模态窗口加载反馈

5.1.6 弹出式操作结果

弹出式提示 (toast) 适用于轻量级的成功提示, 1.5秒后自动消失, 并不打断流程, 对用户影响较小, 适用于不需要强调的操作提醒, 如成功提示, 如图5.8所示。需要特别注意的是, 该形式不适用于错误提示, 因为错误提示需明确告知用户, 因而不适合使用一闪而过的弹出式提示。

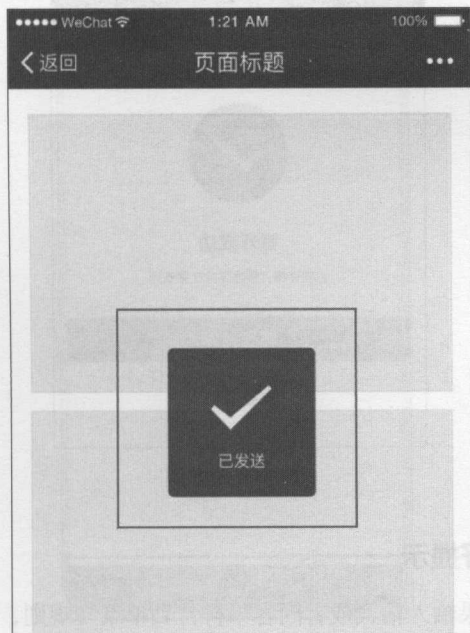


图5.8 弹出式操作结果

5.1.7 模态对话框操作结果

对于需要用户明确知晓的操作结果可通过模态对话框来提示，并可附带下一步操作指引，如图5.9所示。

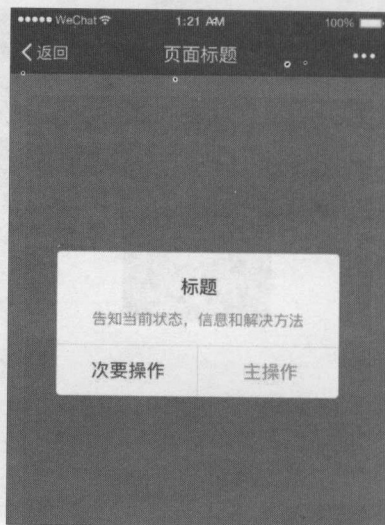


图5.9 模态对话框操作结果

5.1.8 结果页

对于操作结果已经是当前流程的终结的情况，可使用操作结果页来反馈。这种方式最为强烈和明确地告知用户操作已经完成，并可根据实际情况给出下一步操作的指引，如图5.10所示。



图5.10 结果页

5.1.9 表单填写友好提示

用户在填写表单时，如果输入格式或者内容不符合表单填写规则，则需要给用户及时反馈表单填写问题，可以在表单顶部告知错误原因，并标识出错误字段提示用户修改，如图5.11所示。



图5.11 表单友好提示

5.2 微信小程序问答

1 如何将元素固定在界面，不随界面而滚动？

如果界面底部有筛选、出发时间、旅行时间、显示价格4个导航菜单，则把它固定在界面底部，如图5.12所示。

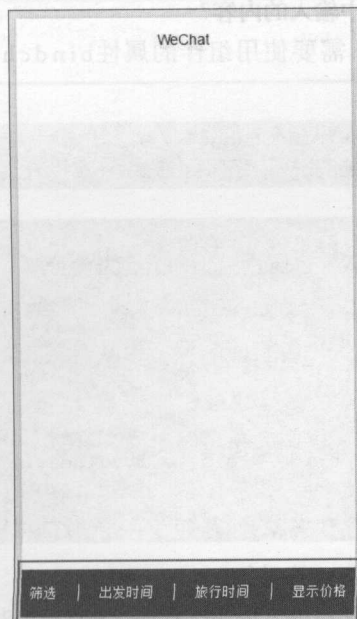


图5.12 将导航菜单固定在界面底部

WXML示例代码如下。

```
<view class="bottomNav">
  <view id="0" class="common" bindtap="switchNav"> 筛选 </view>
  <view style="color:#ffffff">|</view>
  <view id="1" class="common" bindtap="switchNav"> 出发时间 </view>
  <view style="color:#ffffff">|</view>
  <view id="2" class="common" bindtap="switchNav"> 旅行时间 </view>
  <view style="color:#ffffff">|</view>
  <view id="3" class="common" bindtap="switchNav"> 显示价格 </view>
</view>
```

WXSS示例代码如下。

```
.bottomNav{
  background-color: #505963;
  display: flex;
  flex-direction: row;
  height: 45px;
  line-height: 45px;
  position: fixed;
  bottom: 0px;
  width: 100%;
}
.bottomNav view{
  margin: 0 auto;
}
.common{
  font-size: 13px;
  color: #ffffff;
}
```

2 怎样获取用户在表单组件中输入的内容?

能够获取用户输入的组件,需要使用组件的属性bindchange将用户输入的内容同步到AppService。

```
<input id="myInput" bindchange="bindChange" />
<checkbox id="myCheckbox" bindchange="bindChange" />
```

```
var inputContent = {}

Page({
  data: {
    inputContent: {}
  },
  bindChange: function(e) {
    inputContent[e.currentTarget.id] = e.detail.value
  }
})
```

3 为什么脚本内不能使用window等对象?

页面的脚本逻辑是在JsCore中运行的,JsCore是一个没有窗口对象的环境,所以不能在脚本中使用window,也无法在脚本中操作组件。

4 wx.navigateTo无法同时打开超过5个页面吗?

一个应用同时只能打开5个页面,当已经打开了5个页面之后,wx.navigateTo不能正常打开新页面。请避免多层级的交互方式,或者使用wx.redirectTo。

5 如何修改窗口的背景色?

使用 page 标签选择器可以修改顶层节点的样式。

```
page {  
  display: block;  
  min-height: 100%;  
  background-color: red;  
}
```

6 如何在跳转时带参数和在跳转到的界面接收参数?

跳转带参数,示例代码如下。

```
Page({  
  btn: function () {  
    wx.navigateTo({  
      url: '../index/index?id=' + 1000 + "&name=" + kevin  
    })  
  }  
})
```

接收参数,示例代码如下。

```
Page({  
  onLoad: function(e){  
    var id = e.id;  
    var name = e.name;  
  }  
})
```

5.3 小结

本章主要介绍微信小程序在设计过程中遇到的问题以及如何提高用户的体验度,重点掌握以下内容。

- 1 在设计过程中要突出重点,减少干扰项,给用户明确的主次操作提示,让用户操作流程顺畅。
- 2 在用户的操作过程中要及时给予反馈,局部加载以及模态窗口加载都是对用户操作的反馈。
- 3 在用户操作后要给予明确的操作结果,可以通过弹出式操作结果、模态对话框操作结果等将结果内容告知用户。
- 4 在用户填写表单时,要进行友好的提示和正确的引导,以减少用户填写表单的时间和出错的几率。

- 5 理解微信小程序设计过程中遇到的一些问题以及解决方案。

第二篇

综合案例应用

第6章 综合案例：仿智行火车票12306 微信小程序

智行火车票是一款收取费用的用来自动查询预订火车票的软件，可以实时监控票数的多少，与铁道部数据实时同步。通过该软件可以直接在12306上订票，可以查询、预定和购买。它还对12306购票流程进行了大量优化，使用户购票更加快捷。软件还额外提供了智能查询和火车票监控功能。智行火车票App的主要界面如图6.1~图6.4所示。



图6.1 火车票

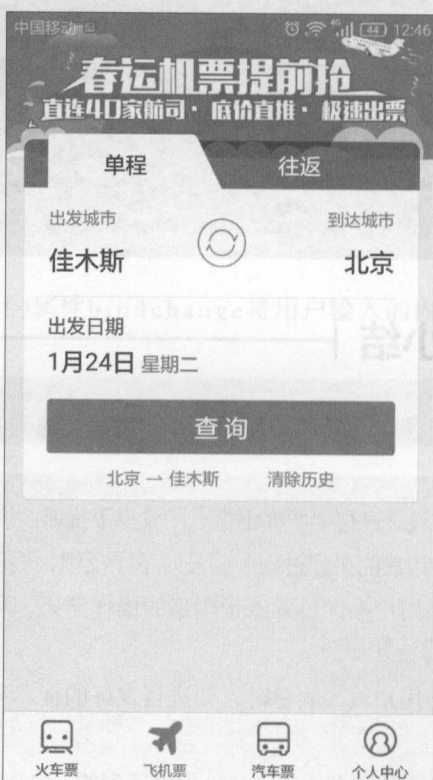


图6.2 飞机票



图6.3 汽车票



图6.4 个人中心

6.1 需求描述

仿智行火车票12306微信小程序要完成以下功能。

1 底部标签导航，放置4个标签：火车票、飞机票、汽车票、个人中心，选中效果图片呈现为蓝色，字体颜色设置为蓝色，如图6.5所示。

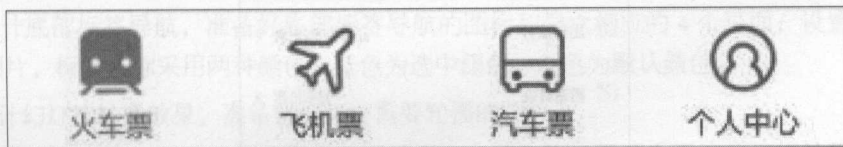


图6.5 标签选中效果

2 在火车票界面中，顶部区域放置海报轮播区域，中间放置火车票、飞机票的查询内容，下面放置快捷导航菜单：极速抢票、在线选座、抢手好货、超值酒店，如图6.6所示。

3 在火车票界面中，输入起始站和终点站，单击“查询”按钮，可以查到火车票信息，包括车站名称、车次、历时时长、发车时间、到站时间、票价，如图6.7所示。



图6.6 火车票界面



图6.7 火车票列表

4 在个人中心界面，设计账号信息内容、订单菜单导航以及二级页面的列表式导航，如图6.8所示。

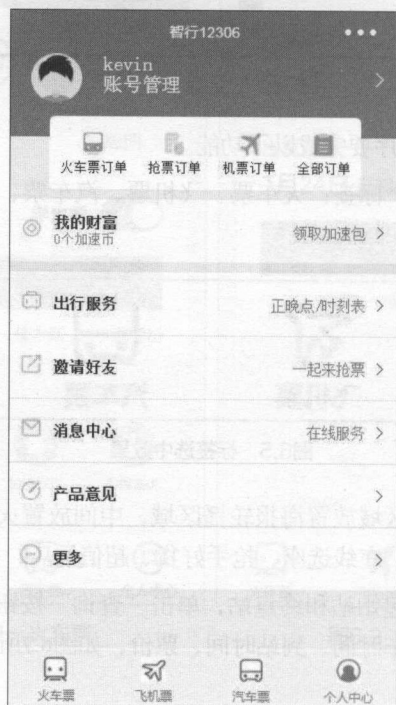


图6.8 个人中心界面

5 在个人中心界面，单击“邀请好友一起来抢票”导航，可查看抢票情况，如图6.9所示。



图6.9 抢票界面

6.2 设计思路及相关知识点

6.2.1 设计思路

1 设计底部标签导航，准备好底部标签导航的图标和建立相应的4个页面；设置默认时图片和选中时图片，标签名称采用两种颜色，蓝色为选中颜色，灰色为默认颜色。

2 设计幻灯片轮播效果，准备好幻灯片需要轮播的图片。

3 设计火车票查询区域，火车票查询区域有火车票和飞机票两个页签的切换效果，切换选中时，背景色为白色，文字颜色为黑色。

4 在火车票列表界面，可以先设计出一条火车票信息，然后发起网络请求获得所有的火车票列表，采用列表渲染的方式展现出所有火车票信息。

5 个人中心采用列表式导航的方式来进行二级界面导航，这种导航设计也是先设计出一个菜单，其他的导航菜单可以直接复用这个菜单的界面效果。

6 抢票界面也是非常有规律性的界面，先设计出第一个区域的内容，其他区域的内容可以直接进行复用。

6.2.2 相关知识点

- 1 在界面布局的时候，会用到微信小程序的组件，包括view视图容器组件、image图片组件、swiper滑块视图容器组件、表单相关组件等。
- 2 对于界面样式设计，需要写一些wxss样式进行界面的美化和渲染。
- 3 需要使用wx.request发起网络请求获取到火车票相关信息，返回json数据，在界面中进行渲染。
- 4 界面跳转需要使用wx.navigateTo这个API接口。

6.3 准备工作

- 1 需要准备一个AppID，如果没有AppID也没有关系，只不过不能再手机上进行项目的预览，但是在开发工具上开发是没有任何问题的。
- 2 底部标签导航的设置需要有选中图片和默认图片，并将其放置在“images/bar”文件夹下，如图6.10所示。

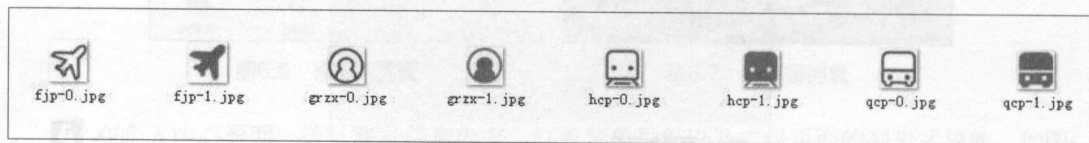


图6.10 底部标签导航图片

- 3 需要准备海报轮播的图片，并将其放置在“images/haibao”文件夹下，如图6.11所示。



图6.11 海报轮播图片

- 4 准备火车票界面要用到的一些图标，并将其放置在“images/icon/hcp”文件夹下，如图6.12所示。

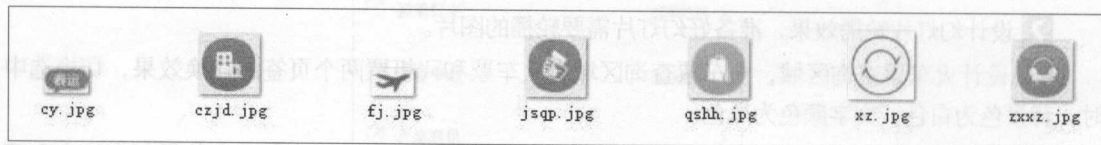


图6.12 火车票界面图标

- 5 准备个人中心界面要用到的一些图标，并将其放置在“images/icon/grzx”文件夹下，如图6.13所示。

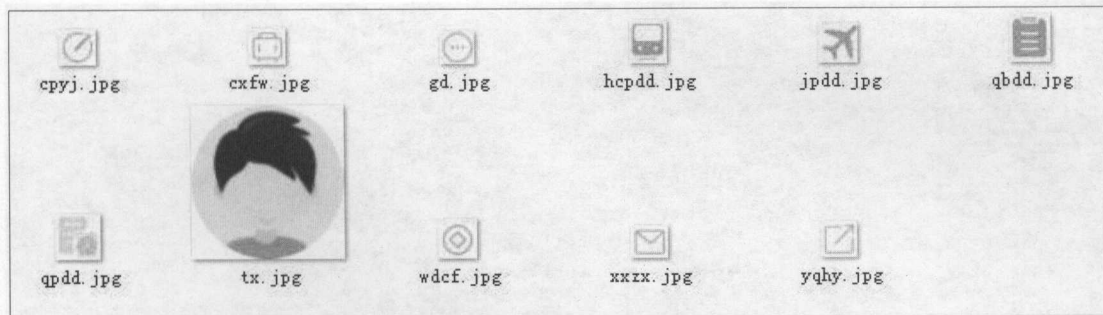


图6.13 个人中心界面图标

6.4 设计流程

仿智行火车票12306微信小程序先来设计底部标签导航，添加导航对应的4个界面：火车票、飞机票、汽车票、个人中心；在火车票界面里设计海报轮播效果、火车票查询界面；再添加一个新的界面——火车票列表界面，在这个界面里完成火车票列表设计；在个人中心界面里完成个人中心界面设计；再添加一个抢票界面，完成抢票界面的显示。

6.4.1 底部标签导航设计

仿智行火车票12306微信小程序，底部标签导航分为4个标签导航：火车票、飞机票、汽车票、个人中心，标签导航选中时导航图标和导航文字都会变为蓝色，如图6.14所示。



图6.14 底部标签导航选中效果

1 新建一个zxtrain项目的微信小程序，将准备好的底部标签导航图标、海报轮播图片、火车票界面图标、个人中心界面图标放置在zxtrain项目下。

2 打开App.json配置文件，在pages数组里添加4个页面路径“pages/train/train”“pages/airplane/airplane”“pages/bus/bus”“pages/mycenter/mycenter”，保存后会自动生成相应的页面文件夹；删除掉“pages/index/index”“pages/logs/logs”页面路径以及对应的文件夹，具体代码如下。

```
{
  "pages": [
    "pages/train/train",
```



```
"pages/airplane/airplane",
"pages/bus/bus",
"pages/mycenter/mycenter"
],
"window":{
  "backgroundTextStyle":"light",
  "navigationBarBackgroundColor": "#fff",
  "navigationBarTitleText": "WeChat",
  "navigationBarTextStyle":"black"
}
```

3 在window数组里配置窗口导航背景颜色为蓝色（#494949），导航栏文字为“智行12306”，字体颜色设置为白色（#ffff），具体代码如下。

```
{
  "pages": [
    "pages/train/train",
    "pages/airplane/airplane",
    "pages/bus/bus",
    "pages/mycenter/mycenter"
  ],
  "window": {
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#5495E6",
    "navigationBarTitleText": "智行 12306",
    "navigationBarTextStyle": "white"
  }
}
```

4 在tabBar对象里配置底部标签导航背景色为白色（#303133），文字默认颜色为灰色，选中时为蓝色（#5495E6），在list数组里配置底部标签导航对应的页面、导航名称、默认时图标、选中时图标，具体代码如下。

```
{
  "pages": [
    "pages/train/train",
    "pages/airplane/airplane",
    "pages/bus/bus",
    "pages/mycenter/mycenter",
    "pages/trainList/trainList",
    "pages/grabticket/grabticket"
  ],
  "window": {
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#5495E6",
    "navigationBarTitleText": "智行 12306",
    "navigationBarTextStyle": "white"
  },
  "tabBar": {
```

```
"selectedColor": "#5495E6",
"backgroundColor": "#ffffff",
"borderStyle": "white",
"list": [{
  "pagePath": "pages/train/train",
  "text": "火车票",
  "iconPath": "images/bar/hcp-0.jpg",
  "selectedIconPath": "images/bar/hcp-1.jpg"
}, {
  "pagePath": "pages/airplane/airplane",
  "text": "飞机票",
  "iconPath": "images/bar/fjp-0.jpg",
  "selectedIconPath": "images/bar/fjp-1.jpg"
}, {
  "pagePath": "pages/bus/bus",
  "text": "汽车票",
  "iconPath": "images/bar/qcp-0.jpg",
  "selectedIconPath": "images/bar/qcp-1.jpg"
}, {
  "pagePath": "pages/mycenter/mycenter",
  "text": "个人中心",
  "iconPath": "images/bar/grzx-0.jpg",
  "selectedIconPath": "images/bar/grzx-1.jpg"
}]
}
```

这样就完成了仿智行火车票12306微信小程序底部标签导航的配置，单击不同的导航图标，可以进行不同的页面的切换，同时导航图标和导航文字会呈现为选中状态，如图6.15所示。

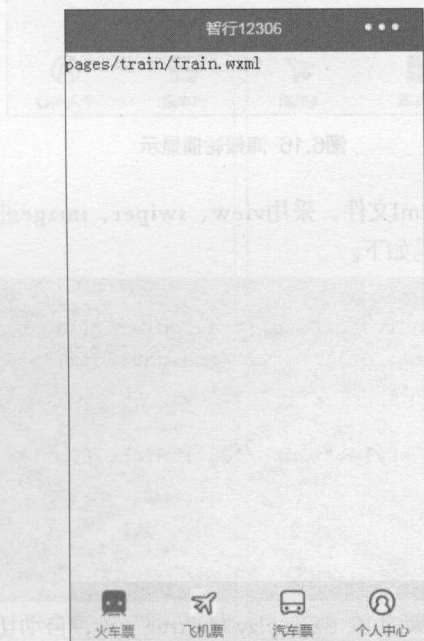


图6.15 火车票界面

6.4.2 海报轮播效果设计

海报轮播效果可以在有限的区域内动态地显示不同的幻灯片图片，是很多网站或者App软件都会采用的一种展现方式，在仿智行火车票12306微信小程序的火车票界面里，采用海报轮播效果展示广告图片，如图6.16所示。

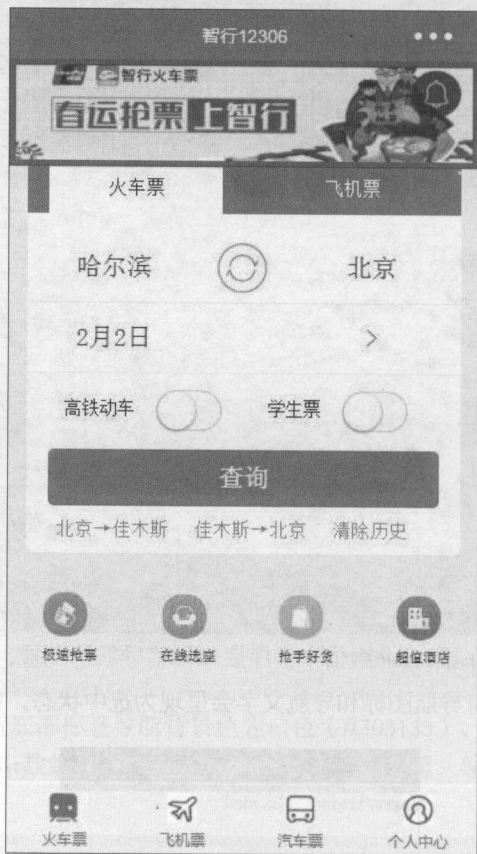


图6.16 海报轮播显示

1 进入pages/train/train.wxml文件，采用view、swiper、image进行布局，将图片宽度设置为100%，高度设置为80px，具体代码如下。

```
<view class="haibao">
  <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval=
    "{{interval}}" duration="{{duration}}" style="height:80px;">
    <block wx:for="{{imgUrls}}">
      <swiper-item>
        <image src="{{item}}" style="width:100%;height:80px;"></image>
      </swiper-item>
    </block>
  </swiper>
</view>
```

swiper滑块视图容器设置为自动播放（autoplay为“true”），自动切换时间间隔为3s（interval为“3000”），滑动动画时长为1s（duration为“1000”）。

采用wx:for循环来显示要展示的图片，从train.js里获取imgUrls图片路径。

2 进入pages/ train/train.js文件，在data对象里定义imgUrls数组，存放海报轮播图片的路径，具体代码如下。

```
Page({
  data: {
    indicatorDots: false,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [
      '/images/haibao/1.jpg',
      '/images/haibao/2.jpg',
      '/images/haibao/3.jpg'
    ]
  },
  onLoad: function (options) {
    // 页面初始化 options 为页面跳转所带来的参数
  }
})
```

这样就可以实现幻灯片轮播效果了，如图6.17和图6.18所示。

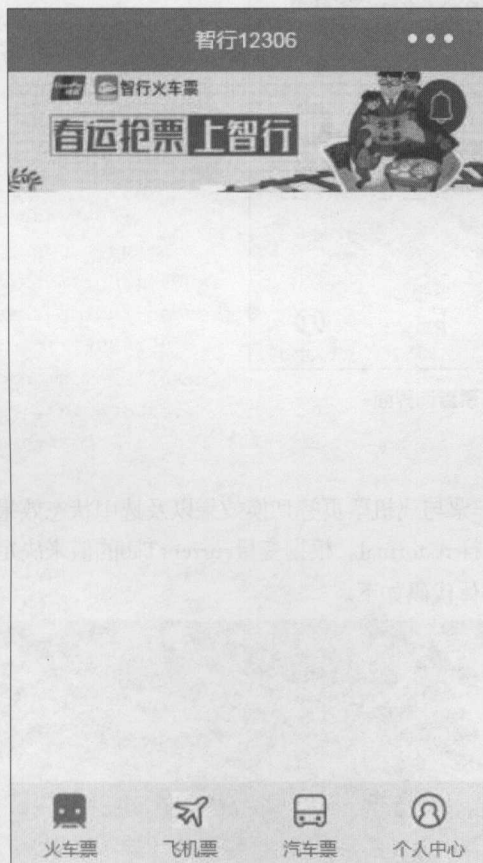


图6.17 海报轮播一

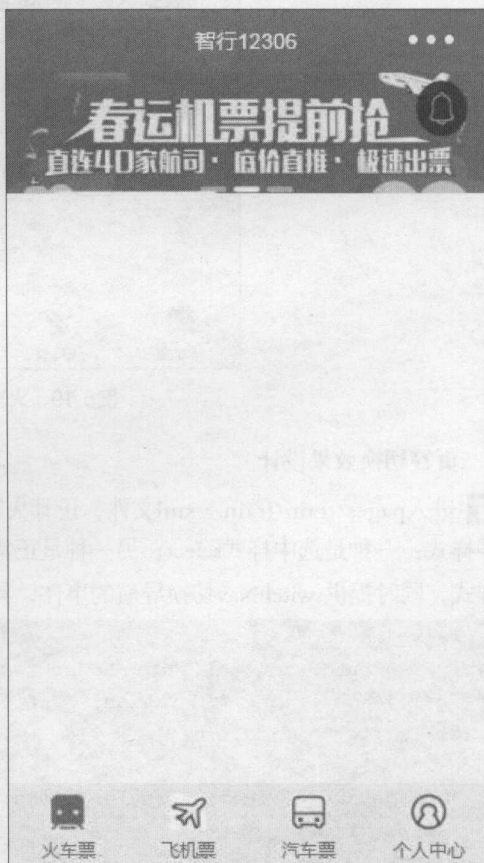


图6.18 海报轮播二

6.4.3 火车票查询界面设计

火车票查询界面提供输入始发站、终点站、出行日期、火车类型等内容进行火车票查询，提供飞机票页签，并与火车票页签进行切换显示；在火车票查询界面下面是4个快捷导航菜单：极速抢票、在线选座、抢手好货、超值酒店，如图6.19所示。

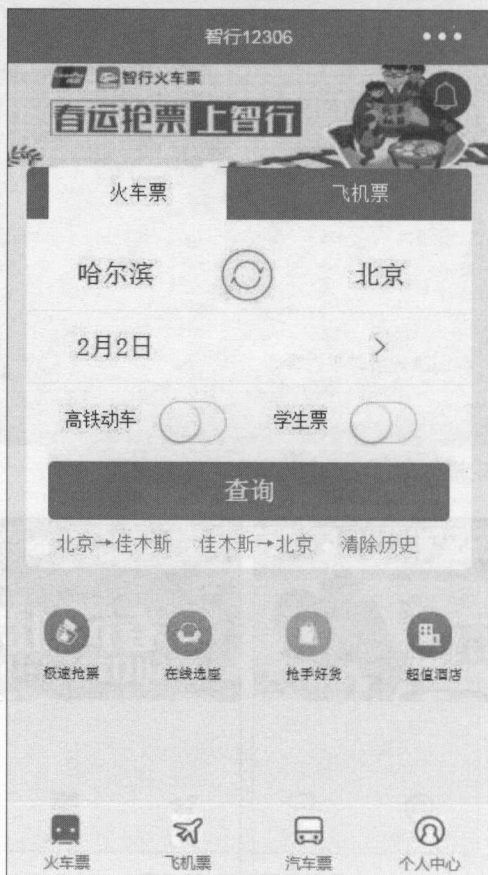


图6.19 火车票查询界面

1. 页签切换效果设计

1 进入pages/train/train.wxml文件，设计火车票与飞机票页签切换效果以及选中状态效果，设计两种样式：一种是选中样式select；另一种是正常样式normal。根据变量currentTab的值来决定使用哪种样式，同时提供switchNav切换导航的事件，具体代码如下。

```
<view class="haibao">
  <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval=
    "{{interval}}" duration="{{duration}}" style="height:80px;">
    <block wx:for="{{imgUrls}}">
      <swiper-item>
        <image src="{{item}}" style="width:100%;height:80px;"></image>
      </swiper-item>
    </block>
  </swiper>
```

```

</view>
<view class="content">
  <view class="navbg">
    <view id="0" class="{{currentTab == 0?'select':'normal'}}" bindtap="switchNav">火
车票</view>
    <view id="1" class="{{currentTab == 1?'select':'normal'}}" bindtap="switchNav">飞
机票</view>
  </view>
</view>

```

2 进入pages/train/train.wxss文件，添加导航背景灰色（#898989）和灰色背景上的圆角矩形；添加页签选中时和默认样式，页签选中时背景色是白色（#ffffff），文字是黑色（#000000），页签默认文字颜色是白色（#fffff），具体代码如下。

```

.content{
  height:500px;
  background-color: #F4F4F4;
}
.navbg{
  width: 92%;
  background-color: #898989;
  height: 40px;
  margin: 0 auto;
  border-top-left-radius:5px;
  border-top-right-radius:5px;
  display: flex;
  flex-direction: row;
}
.select{
  width: 40%;
  height: 40px;
  line-height: 40px;
  text-align: center;
  color: #000000;
  font-size: 15px;
  margin: 0 auto;
  background-color: #ffffff;
}
.normal{
  width: 40%;
  height: 40px;
  line-height: 40px;
  text-align: center;
  color: #ffffff;
  font-size: 15px;
  margin: 0 auto;
}

```

3 进入pages/train/train.js文件，定义变量currentTab的默认值为0，添加switchNav事件，用来进行页签的切换，动态改变变量currentTab的值，具体代码如下。


```
Page({
  data: {
    indicatorDots: false,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrl: [
      '/images/haibao/1.jpg',
      '/images/haibao/2.jpg',
      '/images/haibao/3.jpg'
    ],
    currentTab: 0
  },
  onLoad: function (options) {
    // 页面初始化 options 为页面跳转所带来的参数
  },
  switchNav: function (e) {
    var id = e.currentTarget.id;
    this.setData({ currentTab: id });
  }
})
```

界面效果如图6.20所示。

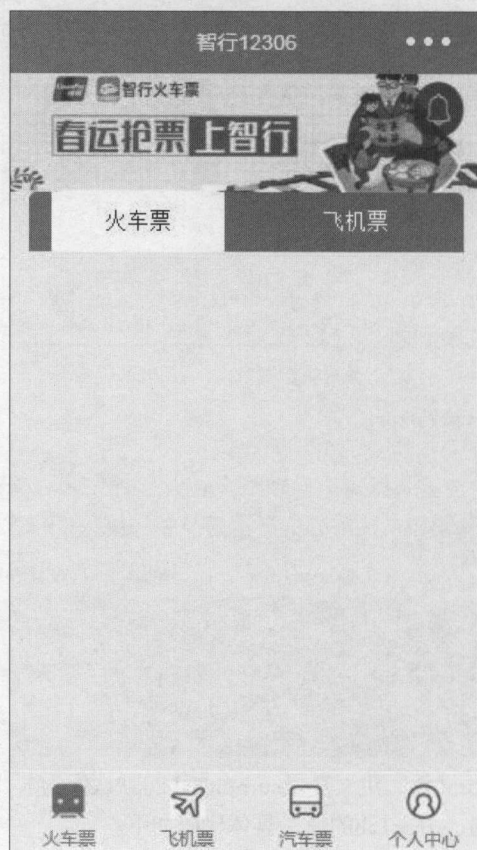


图6.20 页签切换效果

2. 火车票查询区域设计

1 进入pages/train/train.wxml文件，设计火车票查询区域，采用表单组件，input文本框用来输入始发站、终点站、日期，采用switch组件来选择高铁动车票和学生票，提交按钮采用button组件，form组件用来提交表单，具体代码如下。

```
<view class="haibao">
  <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval=
    "{{interval}}" duration="{{duration}}" style="height:80px;">
    <block wx:for="{{imgUrls}}">
      <swiper-item>
        <image src="{{item}}" style="width:100%;height:80px;"></image>
      </swiper-item>
    </block>
  </swiper>
</view>
<view class="content">
  <view class="navbg">
    <view id="0" class="{{currentTab == 0?'select':'normal'}}" bindtap="switchNav">火
    车票</view>
    <view id="1" class="{{currentTab == 1?'select':'normal'}}" bindtap="switchNav">飞
    机票</view>
  </view>
  <view class="formbg">
    <form bindsubmit="formSubmit">
      <view class="station">
        <view>
          <input name="startStation" value=" 哈尔滨 " />
        </view>
        <view>
          <image src="../../images/icon/hcp/xz.jpg" style="width:44px;height:45px;">
</image>
        </view>
        <view>
          <input name="endStation" value=" 北京 " />
        </view>
      </view>
      <view class="hr"></view>
      <view class="station">
        <view>
          <input name="date" value="2月2日" />
        </view>
        <view></view>
        <view>
          <text style="color:#5495E6;">
            <input name="week" value="2月2日" />
          </text><</view>
      </view>
      <view class="hr"></view>
      <view class="type">
        <view> 高铁动车
```

```

        <switch name="gt" type="switch" />
    </view>
    <view> 学生票
        <switch name="xs" type="switch" />
    </view>
</view>
<button class="btn" formType="submit"> 查询 </button>
<view class="record">
    <text> 北京→佳木斯 </text>
    <text> 佳木斯→北京 </text>
    <text> 清除历史 </text>
</view>
</form>
</view>
</view>

```

2 进入pages/train/train.wxss文件，添加相应的样式，具体代码如下。

```

.content{
    height:500px;
    background-color: #F4F4F4;
}
.navbg{
    width: 92%;
    background-color: #898989;
    height: 40px;
    margin: 0 auto;
    border-top-left-radius:5px;
    border-top-right-radius:5px;
    display: flex;
    flex-direction: row;
}
.select{
    width: 40%;
    height: 40px;
    line-height: 40px;
    text-align: center;
    color: #000000;
    font-size: 15px;
    margin: 0 auto;
    background-color: #ffffff;
}
.normal{
    width: 40%;
    height: 40px;
    line-height: 40px;
    text-align: center;
    color: #ffffff;
    font-size: 15px;
    margin: 0 auto;
}
.formbg{

```



```
width: 92%;
background-color: #ffffff;
margin: 0 auto;
padding-top: 20px;
padding-bottom: 10px;
border-bottom-left-radius: 5px;
border-bottom-right-radius: 5px;
}
.station{
  display: flex;
  flex-direction: row;
  width: 90%;
  margin: 0 auto;
  text-align: center;
}
.station view{
  height: 45px;
  line-height: 45px;
  font-size: 20px;
  width: 33%;
}
.station input{
  height: 45px;
  line-height: 45px;
}
.hr{
  height: 1px;
  background-color: #cccccc;
  opacity: 0.2;
  margin-top: 5px;
  margin-bottom: 5px;
}
.type{
  display: flex;
  flex-direction: row;
  width: 90%;
  margin: 0 auto;
  text-align: center;
}
.type view{
  height: 45px;
  line-height: 45px;
  font-size: 14px;
  width: 50%;
}
.type switch{
  margin-left: 10px;
}
.btn{
  width: 90%;
```

```

height: 45px;
line-height: 45px;
color: #ffffff;
text-align: center;
font-size: 20px;
background-color: #5495E6;
margin: 0 auto;
margin-top: 10px;
border-radius: 5px;
}
.record{
text-align: center;
margin-top: 10px;
font-size: 15px;
color: #999999;
}
.record text{
margin-right: 20px;
}

```

界面效果如图6.21所示。

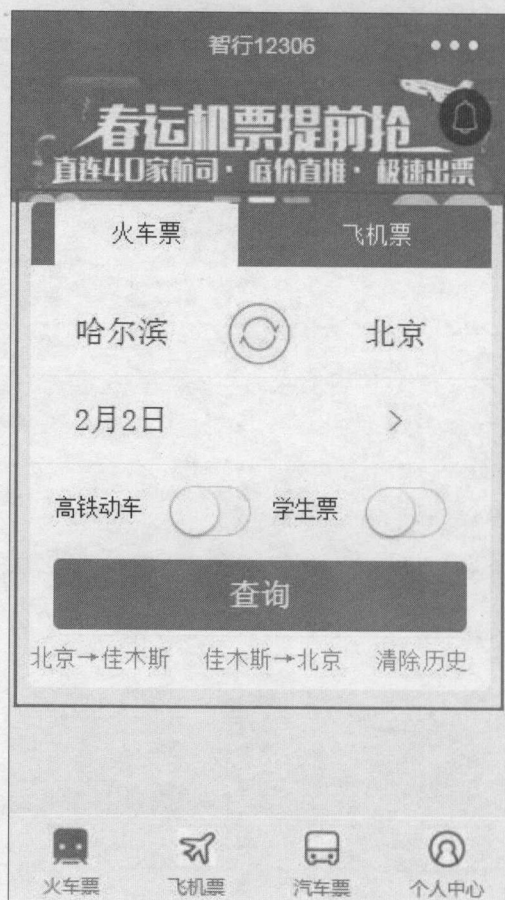


图6.21 火车票查询表单

3 在App.json配置一个新的页面路径“pages/trainList/trainList”，用来设计火车票列表界面，微信小程序框架会自动建立相应的“trainList”文件夹。

4 在pages/train/train.js文件里，添加表单提交formSubmit事件，获得始发站、终点站、日期、星期，把这些数据值带到trainList火车票列表界面，具体代码如下。

```
Page({
  data: {
    indicatorDots: false,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [
      '/images/haibao/1.jpg',
      '/images/haibao/2.jpg',
      '/images/haibao/3.jpg'
    ],
    currentTab: 0
  },
  onLoad: function (options) {
    // 页面初始化 options 为页面跳转所带来的参数
  },
  switchNav: function (e) {
    var id = e.currentTarget.id;
    this.setData({ currentTab: id });
  },
  formSubmit: function (e) {
    console.log(e);
    var startStation = e.detail.value.startStation; // 始发站
    var endStation = e.detail.value.endStation; // 终点站
    var date = e.detail.value.date; // 日期: 2月2日
    var week = e.detail.value.week; // 星期: 周四
    wx.navigateTo({
      url: '../trainList/trainList?startStation=' + startStation + "&endStation=" + endStation + "&date=" + date + "&week=" + week
    })
  }
})
```

在这个火车票查询界面中，输入始发站和终点站，单击“查询”按钮，就会根据输入的内容进行火车票列表的查询。

3. 快捷导航设计

1 进入pages/train/train.wxml文件，在火车票查询界面下面有4个查询按钮，由图标和导航名称组成，具体代码如下。

```
<view class="haibao">
  <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval="{{interval}}"
    duration="{{duration}}" style="height:80px;">
    <block wx:for="{{imgUrls}}">
      <swiper-item>
        <image src="{{item}}" style="width:100%;height:80px;"></image>
      </block>
    </swiper>
  </view>
```



```

        </swiper-item>
      </block>
    </swiper>
  </view>
  <view class="content">
    <view class="navbg">
      <view id="0" class="{{currentTab == 0?'select':'normal'}}" bindtap="switchNav"> 火
车票 </view>
      <view id="1" class="{{currentTab == 1?'select':'normal'}}" bindtap="switchNav"> 飞
机票 </view>
    </view>
    <view class="formbg">
      <form bindsubmit="formSubmit">
        <view class="station">
          <view>
            <input name="startStation" value=" 哈尔滨 " />
          </view>
          <view>
            <image src="../../../images/icon/hcp/xz.jpg" style="width:44px;height:45px;"></
image>
          </view>
          <view>
            <input name="endStation" value=" 北京 " />
          </view>
        </view>
        <view class="hr"></view>
        <view class="station">
          <view>
            <input name="date" value="2月2日" />
          </view>
          <view></view>
          <view>
            <text style="color:#5495E6;">
              <input name="week" value="2月2日" />
            </text><></view>
        </view>
        <view class="hr"></view>
        <view class="type">
          <view> 高铁动车
            <switch name="gt" type="switch" />
          </view>
          <view> 学生票
            <switch name="xs" type="switch" />
          </view>
        </view>
        <button class="btn" formType="submit"> 查询 </button>
        <view class="record">
          <text> 北京→佳木斯 </text>

```

```

        <text> 佳木斯→北京 </text>
        <text> 清除历史 </text>
    </view>
</form>
</view>
<view class="icon">
<view class="item">
    <view>
        <image src="../../../images/icon/hcp/jssp.jpg" style="width:40px;height:40px;">
</image>
    </view>
    <view class="menu"> 极速抢票 </view>
</view>
<view class="item">
    <view>
        <image src="../../../images/icon/hcp/zxxz.jpg" style="width:40px;height:40px;">
</image>
    </view>
    <view class="menu"> 在线选座 </view>
</view>
<view class="item">
    <view>
        <image src="../../../images/icon/hcp/qshh.jpg" style="width:40px;height:40px;">
</image>
    </view>
    <view class="menu"> 抢手好货 </view>
</view>
<view class="item">
    <view>
        <image src="../../../images/icon/hcp/czjd.jpg" style="width:40px;height:40px;">
</image>
    </view>
    <view class="menu"> 超值酒店 </view>
</view>
</view>

```

2 进入pages/train/train.wxss文件，给icon、item、menu这3个class添加样式，具体代码如下。

```

.content{
    height:500px;
    background-color: #F4F4F4;
}
.navbg{
    width: 92%;
    background-color: #898989;
    height: 40px;
    margin: 0 auto;
    border-top-left-radius:5px;

```



```

border-top-right-radius:5px;
display: flex;
flex-direction: row;
}
.select{
width: 40%;
height: 40px;
line-height: 40px;
text-align: center;
color: #000000;
font-size: 15px;
margin: 0 auto;
background-color: #ffffff;
}
.normal{
width: 40%;
height: 40px;
line-height: 40px;
text-align: center;
color: #ffffff;
font-size: 15px;
margin: 0 auto;
}
.formbg{
width: 92%;
background-color: #ffffff;
margin: 0 auto;
padding-top:20px;
padding-bottom: 10px;
border-bottom-left-radius: 5px;
border-bottom-right-radius: 5px;
}
.station{
display: flex;
flex-direction: row;
width: 90%;
margin: 0 auto;
text-align: center;
}
.station view{
height: 45px;
line-height: 45px;
font-size: 20px;
width: 33%;
}
.station input{
height: 45px;
line-height: 45px;

```



```
}  
.hr{  
  height: 1px;  
  background-color: #cccccc;  
  opacity: 0.2;  
  margin-top: 5px;  
  margin-bottom: 5px;  
}  
.type{  
  display: flex;  
  flex-direction: row;  
  width: 90%;  
  margin: 0 auto;  
  text-align: center;  
}  
.type view{  
  height: 45px;  
  line-height: 45px;  
  font-size: 14px;  
  width: 50%;  
}  
.type switch{  
  margin-left: 10px;  
}  
.btn{  
  width: 90%;  
  height: 45px;  
  line-height: 45px;  
  color: #ffffff;  
  text-align: center;  
  font-size: 20px;  
  background-color: #5495E6;  
  margin: 0 auto;  
  margin-top: 10px;  
  border-radius: 5px;  
}  
.record{  
  text-align: center;  
  margin-top: 10px;  
  font-size: 15px;  
  color: #999999;  
}  
.record text{  
  margin-right: 20px;  
}  
.icon{  
  display: flex;  
  flex-direction: row;
```

```
margin-top: 30px;
}
.item{
width: 25%;
text-align: center;
margin: 0 auto;
}
.menu{
font-size: 11px;
width:100px;
}
```

界面效果如图6.22所示。

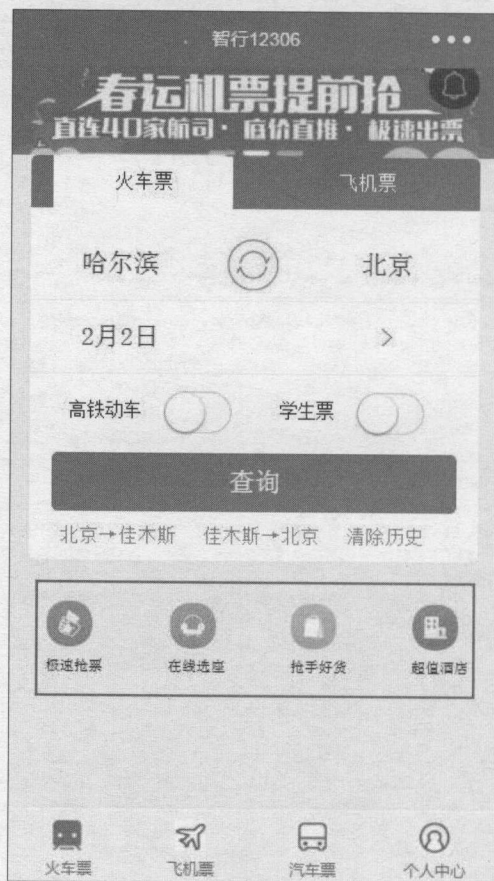


图6.22 快捷导航设计

这样就完成了火车票和飞机票页签的切换效果、火车票查询表单的设计及快捷导航的设计，输入表单内容，就可以根据这些表单内容进行火车票列表的查询了。

6.4.4 火车票列表设计

从火车票查询界面跳转到火车票列表界面，会把始发站、终点站、日期这些查询条件带到火车票列表界面，根据这些查询条件，会加载出相应的火车票信息，如图6.23所示。



图6.23 火车票列表界面

1. 导航标题和日期的展示

在火车票列表的上面是导航标题和日期的展示，导航标题是从上一个界面传递过来的始发站和终点站的参数，日期也是从上一个界面传递过来的参数，如图6.24所示。



图6.24 导航标题和日期

1 进入pages/trainList/trainList.js文件，在onLoad生命周期函数里接收上一个界面传递过来的参数，并通过wx.setNavigationBarTitle来设置导航标题，具体代码如下。

```
Page({
  data:{
  },
  onLoad:function(e){
    var startStation = e.startStation;// 始发站
    var endStation = e.endStation;// 终点站
    var date = e.date;// 日期
    console.log("startStation="+startStation+"---endStation="+endStation+"---date="+date);
    wx.setNavigationBarTitle({
      title: startStation+' → '+endStation
    });
  }
})
```

2 进入pages/trainList/trainList.wxml文件，进行日期的设计，包括查询的日期、前一天和后一天，具体代码如下。

```
<view class="date">
  <view> 前一天</view>
  <view>02月02日周四</view>
  <view> 后一天</view>
</view>
```

3 进入pages/trainList/trainList.wxss文件，给date这个class添加样式，设计成蓝色背景，文字设置成白色（#ffffff），具体代码如下。

```
.date{
  height:40px;
  background-color: #5495E6;
  display: flex;
  flex-direction: row;
}
.date view{
  margin: 0 auto;
  color: #ffffff;
  padding-top: 10px;
}
```

4 进入pages/trainList/trainList.js文件，定义一个日期date变量，把传递过来的日期赋值给date变量，具体代码如下。

```
Page({
  data:{
    date:''
  },
  onLoad:function(e){
    var startStation = e.startStation;// 始发站
    var endStation = e.endStation;// 终点站
    var date = e.date;// 日期
    console.log("startStation="+startStation+"---endStation="+endStation+"---date="+date);
```

```

wx.setNavigationBarTitle({
  title: startStation+'→'+endStation
});
this.setData({data:date});
}
})

```

5 进入ages/trainList/trainList.wxml文件，将date变量绑定到WXML文件里，动态显示日期，具体代码如下。

```

<view class="date">
  <view> 前一天 </view>
  <view>{{date}}</view>
  <view> 后一天 </view>
</view>

```

界面效果如图6.25所示。

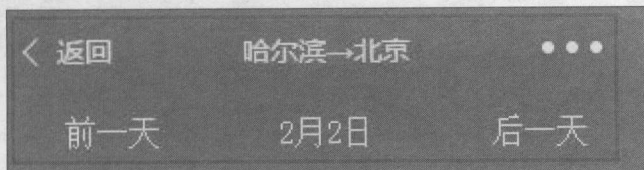


图6.25 动态显示导航标题和日期

2. 火车票列表设计

火车票列表应很有规律地进行展示，每一条火车票信息都包含始发站、终点站、车次、日期、票价等各种信息，所以可以先设计一条火车票信息，然后去请求火车票列表接口，以获取火车票列表信息，最后将信息用列表渲染的方式展现出来。

1 进入pages/trainList/trainList.wxml文件，设计一条火车票信息的布局，具体代码如下。

```

<view class="date">
  <view> 前一天 </view>
  <view>{{date}}</view>
  <view> 后一天 </view>
</view>
<view class="content" style="height:{{winHeight}}px">
  <view class="bg">
    <view class="item">
      <view class="wrApper left">
        <view class="normal"> 哈尔滨 </view>
        <view class="blue"> 北京 </view>
      </view>
      <view class="wrApper center">
        <view class="normal"> DD28 </view>
        <view class="line"></view>
        <view class="small"> 7 小时 54 分 </view>
      </view>
      <view class="wrApper right">
        <view class="normal"> 06:57 </view>
        <view class="normal"> 14:51 </view>
      </view>
    </view>
  </view>
</view>

```

```

</view>
<view class="wrApper right">
  <view class="blue">¥306.5 起</view>
  <view class="buy">可抢票</view>
</view>
</view>
<view class="hr"></view>
<view class="seat">
  <view class="yes">一等座:10 张<text>(抢)</text></view>
  <view class="no">二等座:0 张<text>(抢)</text></view>
</view>
</view>
</view>

```

2 进入pages/trainList/trainList.wxss文件，给火车票信息添加样式，具体代码如下。

```

.date{
  height:40px;
  background-color: #5495E6;
  display: flex;
  flex-direction: row;
}
.date view{
  margin: 0 auto;
  color: #ffffff;
  padding-top: 10px;
}
.content{
  height:600px;
  background-color: #F4F4F4;
  padding-top:10px;
}
.bg{
  width: 95%;
  height: 90px;
  background-color: #ffffff;
  margin: 0 auto;
  border-radius: 5px;
  margin-bottom: 10px;
}
.item{
  display: flex;
  flex-direction: row;
  padding: 10px;
}
.wrApper{
  width: 25%;
}
.left{
  text-align: left;
}

```



```
.center{
    text-align: center;
}
.right{
    text-align: right;
}
.blue{
    color: #5495E6;
    font-weight: bold;
    font-size: 16px;
}
.normal{
    color: #000000;
    font-weight: bold;
    font-size: 16px;
}
.small{
    font-size: 13px;
    color: #666666;
}
.line{
    height: 1px;
    background-color: #cccccc;
    opacity: 0.2;
    width: 80%;
    margin: 0 auto;
    margin-top: 3px;
    margin-bottom: 3px;
}
.buy{
    background-color: red;
    width: 42px;
    height: 20px;
    line-height: 20px;
    font-size: 12px;
    color: #ffffff;
    text-align: center;
    float: right;
    border-radius: 20px;
    margin-right: 5px;
}
.hr{
    height: 1px;
    background-color: #cccccc;
    opacity: 0.2;
}
.seat{
    font-size: 13px;
    display: flex;
    flex-direction: row;
```

```
margin-top: 5px;
margin-left: 10px;
}
.seat text{
  color: red;
}
.no{
  color: #999999;
  margin-right: 10px;
}
.yes{
  margin-right: 10px;
}
```

界面效果如图6.26所示。

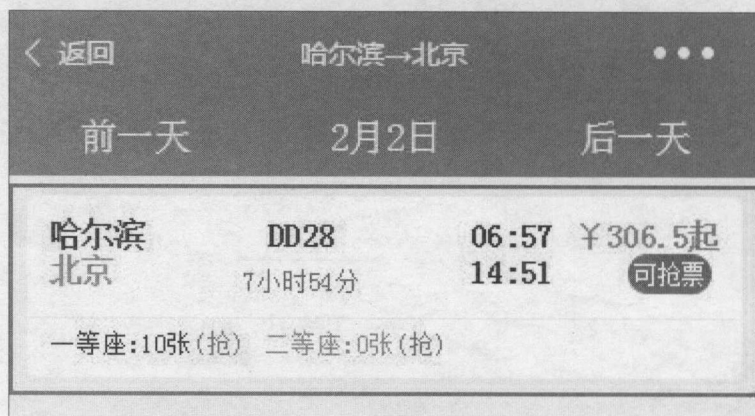


图6.26 火车票信息

3 进入pages/trainList/trainList.js文件，去动态地获取火车票列表信息，定义trainList变量用来绑定火车票列表信息，定义winHeight变量来动态地计算列表信息，具体代码如下。

```
Page({
  data: {
    date: '',
    trainList: [],
    winHeight: 600
  },
  onLoad: function(e) {
    var startStation = e.startStation; // 始发站
    var endStation = e.endStation; // 终点站
    var date = e.date; // 日期
    console.log("startStation="+startStation+"---endStation="+endStation+"---date="+date);
    wx.setNavigationBarTitle({
      title: startStation+' → '+endStation
    });
    this.setData({date:date});
    this.loadTrainsList(startStation,endStation);
  },

```

```

loadTrainsList:function(startStation,endStation){
    var page = this;
    wx.request({
        url: 'https://apis.juhe.cn/train/s2swithprice',
        data: {
            start:startStation,// 始发站
            end:endStation,// 终点站
            date:'',// 日期,如果不填,默认查询明天的火车票信息
            dtype:'',// 火车类型
            key:'5a54c3ff25d0b336ff7dad3d4e889c65'
        },
        method: 'GET',
        success: function(res){
            console.log(res);
            var trainList = res.data.result.list;
            var trainList = wx.getStorageSync('trainList');
            var size = trainList.length;
            var winHeight = size * 100 + 30;
            page.setData({trainList:trainList});
            page.setData({winHeight:winHeight});
        }
    });
}
}
})

```

trainList数据结构如图6.27所示。

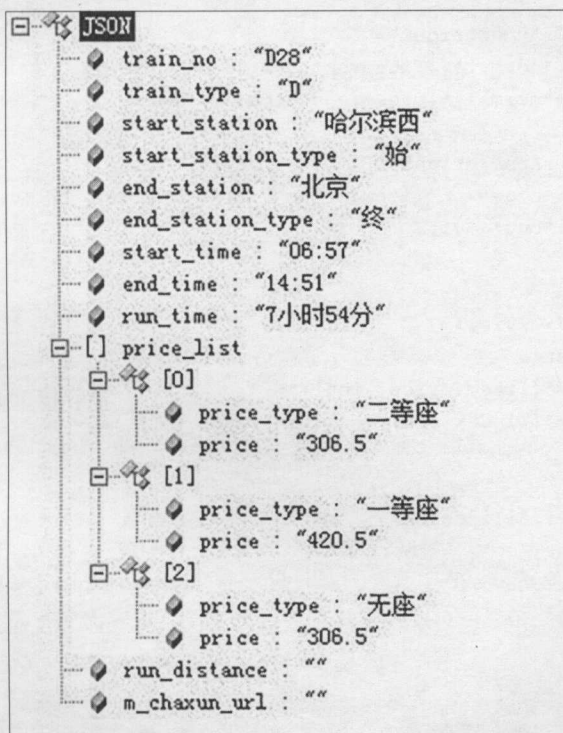


图6.27 trainList数据结构



注意：<https://apis.juhe.cn/train/s2swithprice>这个火车票数据接口有请求次数的限制，一旦请求次数到达限制，就不能返回数据，可以自行去<https://www.juhe.cn/>网站注册账号，免费获取数据接口。

4 进入pages/trainList/trainList.wxml文件，将trainList变量、winHeight变量动态地绑定到界面里，具体代码如下。

```
<view class="date">
  <view> 前一天 </view>
  <view>{{date}}</view>
  <view> 后一天 </view>
</view>
<view class="content" style="height:{{winHeight}}px">
  <block wx:for="{{trainList}}">
    <view class="bg">
      <view class="item">
        <view class="wrApper left">
          <view class="normal">{{item.start_station}}</view>
          <view class="blue">{{item.start_time}}</view>
        </view>
        <view class="wrApper center">
          <view class="normal">{{item.train_no}}</view>
          <view class="line"></view>
          <view class="small">{{item.run_time}}</view>
        </view>
        <view class="wrApper right">
          <view class="normal">{{item.end_station}}</view>
          <view class="normal">{{item.end_time}}</view>
        </view>
        <view class="wrApper right">
          <view class="blue">¥{{item.price_list[0].price}} 起</view>
          <view class="buy">可抢票</view>
        </view>
      </view>
    </view>
    <view class="hr"></view>
    <view class="seat">
      <block wx:for="{{item.price_list}}">
        <view wx:if="{{index == 0}}" class="yes">二等座:100 张
          <text>( 抢 )</text>
        </view>
        <view wx:elif="{{index < 3}}">
          <view class="no">二等座:0 张
            <text>( 抢 )</text>
          </view>
        </view>
      </block>
    </view>
  </view>
</block>
```

</view>

界面效果如图6.28所示。



图6.28 火车票列表

设计列表或者有规律布局的界面时，可以先设计共有的区域内容作为基础单元，然后复制或者列表循环展示这块共有的内容，就像火车票列表信息一样。

3. 火车票底部固定页签导航

火车票列表界面最底部的区域是一块固定页签导航，它不随界面的滚动而滚动，而是固定在底部，有4个页签导航：筛选、出发时间、旅行时间、价格，根据这些页签导航可进行火车票列表的展示。

1 进入pages/trainList/trainList.wxml文件，设计底部固定导航，需要设计两种样式：一种是导航标题选中效果select，文字呈现为蓝色；另一种是默认样式效果common，定义变量currentTab来动态地渲染样式，定义switchNav菜单来切换事件，具体代码如下。

```
<view class="date">
  <view> 前一天 </view>
  <view>{{date}}</view>
```

```

    <view>后一天</view>
  </view>
  <view class="content" style="height:{{winHeight}}px">
    <block wx:for="{{trainList}}">
      <view class="bg">
        <view class="item">
          <view class="wrApper left">
            <view class="normal">{{item.start_station}}</view>
            <view class="blue">{{item.start_time}}</view>
          </view>
          <view class="wrApper center">
            <view class="normal">{{item.train_no}}</view>
            <view class="line"></view>
            <view class="small">{{item.run_time}}</view>
          </view>
          <view class="wrApper right">
            <view class="normal">{{item.end_station}}</view>
            <view class="normal">{{item.end_time}}</view>
          </view>
          <view class="wrApper right">
            <view class="blue">¥{{item.price_list[0].price}} 起</view>
            <view class="buy">可抢票</view>
          </view>
        </view>
      </view>
      <view class="hr"></view>
      <view class="seat">
        <block wx:for="{{item.price_list}}">
          <view wx:if="{{index ==0}}" class="yes">二等座:100 张
            <text>( 抢 )</text>
          </view>
          <view wx:elif="{{index < 3}}">
            <view class="no">二等座:0 张
              <text>( 抢 )</text>
            </view>
          </view>
        </block>
      </view>
    </view>
  </block>
  <view class="bottomNav">
    <view id="0" class="{{currentTab==0?'selected':'common'}}" bindtap="switchNav">
      筛选</view>
    <view style="color:#ffffff">|</view>
    <view id="1" class="{{currentTab==1?'selected':'common'}}" bindtap="switchNav">
      出发时间</view>
    <view style="color:#ffffff">|</view>
    <view id="2" class="{{currentTab==2?'selected':'common'}}" bindtap="switchNav">
      旅行时间</view>
    <view style="color:#ffffff">|</view>
    <view id="3" class="{{currentTab==3?'selected':'common'}}" bindtap="switchNav">

```


显示价格 </view>

</view>

</view>

2 进入pages/trainList/trainList.wxss文件，添加页签导航的选中效果样式和默认效果样式，具体代码如下。

```
.date{
  height:40px;
  background-color: #5495E6;
  display: flex;
  flex-direction: row;
}
.date view{
  margin: 0 auto;
  color: #ffffff;
  padding-top: 10px;
}
.content{
  height:600px;
  background-color: #F4F4F4;
  padding-top:10px;
}
.bg{
  width: 95%;
  height: 90px;
  background-color: #ffffff;
  margin: 0 auto;
  border-radius: 5px;
  margin-bottom: 10px;
}
.item{
  display: flex;
  flex-direction: row;
  padding: 10px;
}
}
.wrApper{
  width: 25%;
}
}
.left{
  text-align: left;
}
}
.center{
  text-align: center;
}
}
.right{
  text-align: right;
}
}
.blue{
  color: #5495E6;
```

```

    font-weight: bold;
    font-size: 16px;
}
.normal{
    color: #000000;
    font-weight: bold;
    font-size: 16px;
}
.small{
    font-size: 13px;
    color: #666666;
}
.line{
    height: 1px;
    background-color: #cccccc;
    opacity: 0.2;
    width: 80%;
    margin: 0 auto;
    margin-top: 3px;
    margin-bottom: 3px;
}
.buy{
    background-color: red;
    width: 42px;
    height: 20px;
    line-height: 20px;
    font-size: 12px;
    color: #ffffff;
    text-align: center;
    float: right;
    border-radius: 20px;
    margin-right: 5px;
}
.hr{
    height: 1px;
    background-color: #cccccc;
    opacity: 0.2;
}
.seat{
    font-size: 13px;
    display: flex;
    flex-direction: row;
    margin-top: 5px;
    margin-left: 10px;
}
.seat text{
    color: red;
}
.no{
    color: #999999;

```

```

margin-right:10px;
}
.yes{
margin-right:10px;
}
.bottomNav{
background-color: #505963;
display: flex;
flex-direction: row;
height: 45px;
line-height: 45px;
position: fixed;
bottom:0px;
width: 100%;
}
.bottomNav view{
margin: 0 auto;
}
.selected{
font-size: 13px;
color: #5495E6;
}
.common{
font-size: 13px;
color: #ffffff;
}

```

3 进入pages/trainList/trainList.js文件，定义变量currentTab的值为1，添加菜单切换事件switchNav，用来动态地设置页签导航选中效果，具体代码如下。

```

Page({
  data:{
    date:'',
    trainList:[],
    winHeight:600,
    currentTab:'1'
  },
  onLoad:function(e){
    var startStation = e.startStation;// 始发站
    var endStation = e.endStation;// 终点站
    var date = e.date;// 日期
    console.log("startStation="+startStation+"---endStation="+endStation+"---date="+date);
    wx.setNavigationBarTitle({
      title: startStation+' → '+endStation
    });
    this.setData({date:date});
    this.loadTrainsList(startStation,endStation);
  },
  loadTrainsList:function(startStation,endStation){
    var page = this;
    wx.request({

```



```
url: 'https://apis.juhe.cn/train/s2swithprice',
data: {
  start:startStation,// 始发站
  end:endStation,// 终点站
  date:'',// 日期, 如果不填, 默认查询明天的火车票信息
  dtype:'',// 火车类型
  key:'5a54c3ff25d0b336ff7dad3d4e889c65'
},
method: 'GET',
success: function(res){
  console.log(res);
  var trainList = res.data.result.list;
  var size = trainList.length;
  var winHeight = size * 100 + 30;
  page.setData({trainList:trainList});
  page.setData({winHeight:winHeight});
}
});
},
switchNav:function (e) {
  var id = e.currentTarget.id;
  console.log(id);
  this.setData({ currentTab: id });
}
})
```

界面效果如图6.29所示。



图6.29 固定页签导航

6.4.5 个人中心界面设计

个人中心界面用来显示账号相关信息、订单情况、我的财富、出行服务、邀请好友、消息中心、产品意见等内容，它是通过列表式导航的方式来设计的，如图6.30所示。

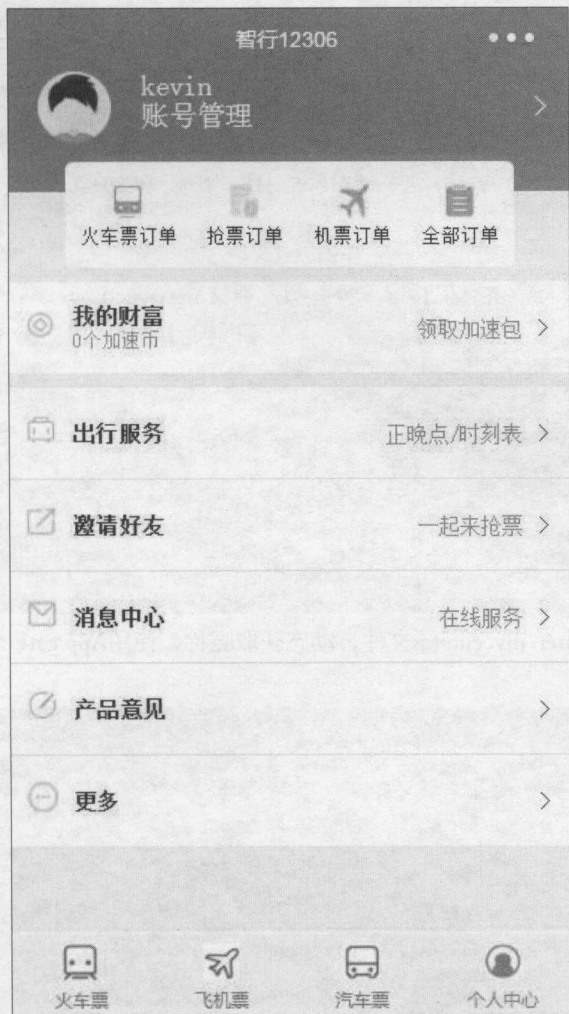


图6.30 个人中心界面

1. 账号信息设计

1 进入pages/mycenter/mycenter.wxml文件，设计账号的头像、昵称等信息，使用变量userInfo作为个人信息来展示，具体代码如下。

```
<view class="amountBg">
  <view class="img"><image src="../../../images/icon/grzx/tx.jpg" style="width:49px;height:47px;">
</image></view>
  <view class="account">
    <view>{{userInfo.nickName}}</view>
    <view> 账号管理 </view>
  </view>
```

```
<view class="nav">></view>
</view>
```

2 进入pages/mycenter/mycenter.wxss文件，给头像、账号、二级界面导航入口添加样式，具体代码如下。

```
.amountBg{
  height:100px;
  background-color: #5495E6;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.img{
  margin-left: 20px;
}
.account{
  width: 70%;
  color: #ffffff;
  margin-left: 10px;
}
.nav{
  width:15px;
  color: #ffffff;
}
```

3 进入pages/mycenter/mycenter.js文件，动态获取昵称，使用App.getUserInfo接口来获取个人信息，具体代码如下。

```
var App = getApp()
Page({
  data:{
    userInfo: {}
  },
  onLoad:function(options){
    var that = this
    // 以调用应用实例的方法获取全局数据
    App.getUserInfo( function(userInfo) {
      console.log(userInfo);
      // 更新数据
      that.setData( {
        userInfo: userInfo
      })
    })
  }
})
```

界面效果如图6.31所示。

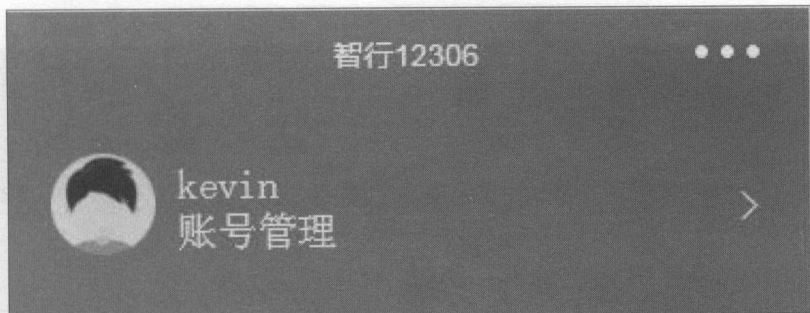


图6.31 账号信息

2. 订单导航设计

1 进入pages/mycenter/mycenter.wxml文件，设计火车票订单、抢票订单、机票订单、全部订单导航，它是由图标和文字组成的，具体代码如下。

```
<view class="amountBg">
  <view class="img">
    <image src="../../../images/icon/grzx/tx.jpg" style="width:49px;height:47px;"></image>
  </view>
  <view class="account">
    <view>{{userInfo.nickName}}</view>
    <view> 账号管理 </view>
  </view>
  <view class="nav">></view>
</view>
<view class="content">
  <view class="order">
    <view class="desc">
      <view><image src="../../../images/icon/grzx/hcpdd.jpg" style="width:22px;height:25px;"></image></view>
      <view> 火车票订单 </view>
    </view>
    <view class="desc">
      <view><image src="../../../images/icon/grzx/qpdd.jpg" style="width:22px;height:25px;"></image></view>
      <view> 抢票订单 </view>
    </view>
    <view class="desc">
      <view><image src="../../../images/icon/grzx/jpdd.jpg" style="width:22px;height:25px;"></image></view>
      <view> 机票订单 </view>
    </view>
    <view class="desc">
      <view><image src="../../../images/icon/grzx/qbdd.jpg" style="width:22px;height:25px;"></image></view>
      <view> 全部订单 </view>
    </view>
  </view>
</view>
```

2 进入pages/mycenter/mycenter.wxss文件，给订单区域添加样式，具体代码如下。

```
.amountBg{
  height:100px;
  background-color: #5495E6;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.img{
  margin-left: 20px;
}
.account{
  width: 70%;
  color: #ffffff;
  margin-left: 10px;
}
.nav{
  width:15px;
  color: #ffffff;
}
.content{
  background-color: #F4F4F4;
  height: 500px;
}
.order{
  width: 94%;
  height: 70px;
  display: flex;
  flex-direction: row;
  background-color: #ffffff;
  border-radius: 5px;
  text-align: center;
  align-items: center;
  position: absolute;
  top:90px;
  margin-left: 3%;
}
.desc{
  width: 25%;
  font-size: 13px;
}
```

界面效果如图6.32所示。

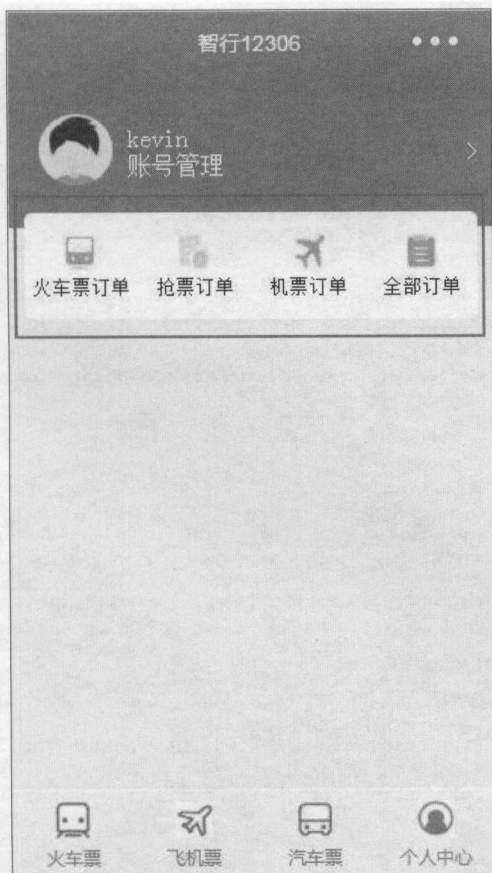


图6.32 订单信息

3. 列表导航设计

1 进入pages/mycenter/mycenter.wxml文件，设计“我的财富”“出行服务”“邀请好友”“消息中心”“产品意见”更多6个列表导航，具体代码如下。

```
<view class="amountBg">
  <view class="img">
    <image src="../../../images/icon/grzx/tx.jpg" style="width:49px;height:47px;"></image>
  </view>
  <view class="account">
    <view>{{userInfo.nickName}}</view>
    <view> 账号管理 </view>
  </view>
  <view class="nav">>></view>
</view>
<view class="content">
  <view class="order">
    <view class="desc">
      <view><image src="../../../images/icon/grzx/hcpdd.jpg" style="width:22px;height:25px;"></image></view>
      <view> 火车票订单 </view>
    </view>
```



```

        <view class="desc">
            <view><image src="../../../images/icon/grzx/qbdd.jpg" style="width:22px;height:
25px;"></image></view>
            <view> 抢票订单 </view>
        </view>
        <view class="desc">
            <view><image src="../../../images/icon/grzx/jpdd.jpg" style="width:22px;height:
25px;"></image></view>
            <view> 机票订单 </view>
        </view>
        <view class="desc">
            <view><image src="../../../images/icon/grzx/qbdd.jpg" style="width:22px;height:
25px;"></image></view>
            <view> 全部订单 </view>
        </view>
    </view>
    <view class="clear"></view>
    <view class="item">
        <view class="icon"><image src="../../../images/icon/grzx/wdcf.jpg" style="width:22px;height:
21px;"></image></view>
        <view class="itemName">
            <view> 我的财富 </view>
            <view class="remark">0 个加速币 </view>
        </view>
        <view class="right"><text class="opr"> 领取加速包 </text><</view>
    </view>
    <view class="line"></view>
    <view class="item">
        <view class="icon"><image src="../../../images/icon/grzx/cxfw.jpg" style="width:22px;height:
21px;"></image></view>
        <view class="itemName">
            <view> 出行服务 </view>
        </view>
        <view class="right"><text class="opr"> 正晚点 / 时刻表 </text><</view>
    </view>
    <view class="hr"></view>
    <view class="item">
        <view class="icon"><image src="../../../images/icon/grzx/yqhy.jpg" style="width:22px;height:
21px;"></image></view>
        <view class="itemName">
            <view> 邀请好友 </view>
        </view>
        <view class="right" bindtap="grabTicket"><text class="opr"> 一起来抢票 </text><</view>
    </view>
    <view class="hr"></view>
    <view class="item">
        <view class="icon"><image src="../../../images/icon/grzx/xxzx.jpg" style="width:22px;height:
21px;"></image></view>
        <view class="itemName">
            <view> 消息中心 </view>
        </view>
    </view>

```

```

</view>
<view class="right"><text class="opr"> 在线服务 </text></view>
</view>
<view class="hr"></view>
<view class="item">
<view class="icon"><image src="../../../images/icon/grzx/cpyj.jpg" style="width:22px;height:
21px;"></image></view>
<view class="itemName">
<view> 产品意见 </view>
</view>
<view class="right"></view>
</view>
<view class="hr"></view>
<view class="item">
<view class="icon"><image src="../../../images/icon/grzx/gd.jpg" style="width:22px;height:
21px;"></image></view>
<view class="itemName">
<view> 更多 </view>
</view>
<view class="right"></view>
</view>
</view>

```

2 进入pages/mycenter/mycenter.wxss文件，给“我的财富”“出行服务”“邀请好友”“消息中心”“产品意见”“更多”这6个列表导航添加样式，具体代码如下。

```

.amountBg{
  height:100px;
  background-color: #5495E6;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.img{
  margin-left: 20px;
}
.account{
  width: 70%;
  color: #ffffff;
  margin-left: 10px;
}
.nav{
  width:15px;
  color: #ffffff;
}
.content{
  background-color: #F4F4F4;
  height: 500px;
}
.order{
  width: 94%;

```



```

height: 70px;
display: flex;
flex-direction: row;
background-color: #ffffff;
border-radius: 5px;
text-align: center;
align-items: center;
position: absolute;
top: 90px;
margin-left: 3%;
}
.desc{
width: 25%;
font-size: 13px;
}
.clear{
padding-top: 70px;
}
.item{
background-color: #ffffff;
display: flex;
flex-direction: row;
height: 50px;
align-items: center;
}
.icon{
width: 50px;
text-align: center;
}
.itemName{
width: 40%;
font-size: 14px;
font-weight: bold;
}
.line{
height: 10px;
}
.hr{
height: 1px;
background-color: #cccccc;
opacity: 0.2;
}
.remark{
font-weight: normal;
margin-top: 5px;
}
.right{
width: 40%;
text-align: right;
}

```



```
.opr{
  color: #5495E6;
  font-size: 13px;
  font-weight: bold;
  margin-right: 10px;
}
```

界面效果如图6.33所示。

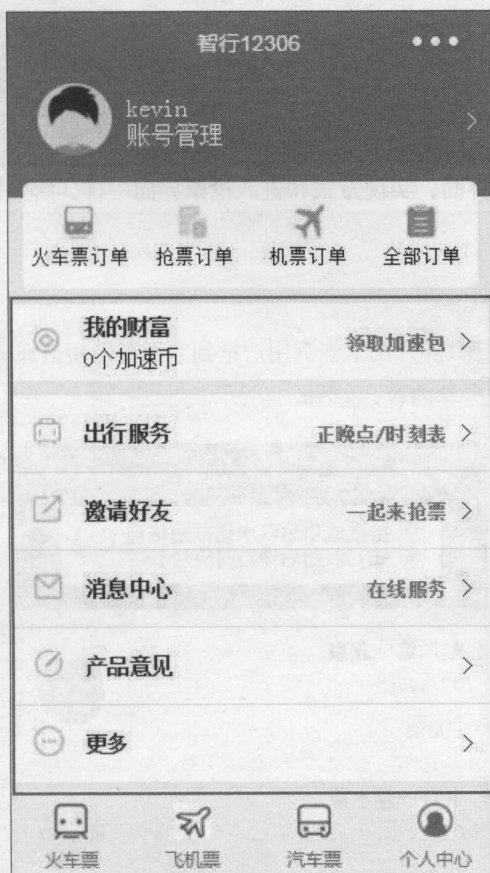


图6.33 个人中心界面

3 给“邀请好友”添加grabTicket抢票事件，单击这个导航会进入抢票界面，查看自己的抢票情况，进入pages/mycenter/mycenter.js文件，绑定grabTicket事件，让它跳转到grabticket界面，在App.json配置grabticket路径，具体代码如下。

```
var App = getApp()
Page({
  data:{
    userInfo: {}
  },
  onLoad:function(options){
    var that = this
    // 调用应用实例的方法获取全局数据
    App.getUserInfo( function(userInfo) {
```

```

console.log(userInfo);
// 更新数据
that.setData({
  userInfo: userInfo
})
}),
grabTicket:function(){
  wx.navigateTo({
    url: '../grabticket/grabticket'
  })
}
})

```

通过列表导航进入二级界面，实现方式和进入抢票界面一样，绑定跳转事件，配置相应的跳转路径即可。

6.4.6 抢票界面设计

抢票界面分为3个部分：第1部分用来告诉用户抢到票后会以短信或者电话的方式进行通知；第2部分是抢票情况；第3部分是分享内容，如图6.34所示。



图6.34 抢票界面

1. 通知区域设计

1 进入pages/ grabticket /grabticket.wxml文件，设计抢票通知区域的内容，包括攻略、抢票通知信息、二级界面入口，具体代码如下。

```
<view class="amountBg">
  <view class="bg">
    <view class="icon"> 攻略 </view>
    <view class="tip"> 抢票成功后以电话或短信的方式通知，收到后请及时支付! </view>
    <view class="right">></view>
  </view>
</view>
```

2 进入pages/ grabticket /grabticket.wxss文件，给通知区域添加样式，设置为蓝色背景，在蓝色背景上添加一块矩形区域用来放置通知区域的内容，具体代码如下。

```
.amountBg{
  height:70px;
  background-color: #5495E6;
}
.bg{
  width: 80%;
  height: 53px;
  background-color: #ffffff;
  border-radius: 5px;
  margin: 0 auto;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.icon{
  border: 1px solid #6EBAFE;
  font-size: 13px;
  margin-left: 10px;
  color: #6EBAFE;
  text-align: center;
}
.tip{
  font-size: 15px;
  margin: 5px;
}
.right{
  margin-right: 5px;
}
```

界面效果如图6.35所示。

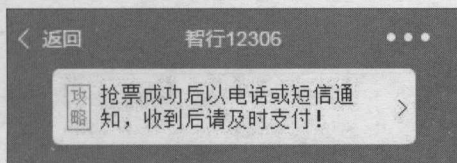


图6.35 通知区域

2. 抢票信息设计

1 进入pages/ grabticket /grabticket.wxml文件，设计已经取消的抢票情况或者可以重新开始抢票，具体代码如下。

```
<view class="amountBg">
  <view class="bg">
    <view class="icon"> 攻略 </view>
    <view class="tip"> 抢票成功后以电话或短信的方式通知，收到后请及时支付！ </view>
    <view class="right"><</view>
  </view>
</view>
<view class="content">
<view class="hr"></view>
  <view class="item">
    <view class="ticket">
      <view class="station"> 哈尔滨→北京 </view>
      <view class="desc1">02月03日 </view>
      <view class="desc2">已取消 </view>
    </view>
    <view class="opr">
      查看
    </view>
  </view>
<view class="hr"></view>
<view class="item">
  <view class="ticket">
    <view class="station"> 北京→佳木斯 </view>
    <view class="desc1">02月07日 </view>
    <view class="desc2">已取消 </view>
  </view>
  <view class="opr">
    查看
  </view>
</view>
<view class="hr"></view>
<view class="item">
  <view class="ticket">
    <view class="station"> 北京→南京 </view>
    <view class="desc1">02月06日 </view>
    <view class="desc2">已取消 </view>
  </view>
  <view class="start">
    开始
  </view>
</view>
<view class="hr"></view>
<view class="item">
  <view class="ticket">
```

```

<view class="station"> 合肥→北京 </view>
<view class="desc1">02月07日 </view>
<view class="desc2">抢票终止 </view>
</view>
<view class="start">
  开始
</view>
</view>
</view>

```

2 进入pages/ grabticket /grabticket.wxss文件，添加相应的样式，具体代码如下。

```

.amountBg{
  height:70px;
  background-color: #5495E6;
}
.bg{
  width: 80%;
  height: 53px;
  background-color: #ffffff;
  border-radius: 5px;
  margin: 0 auto;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.icon{
  border: 1px solid #6EBAFE;
  font-size: 13px;
  margin-left: 10px;
  color:#6EBAFE;
  text-align: center;
}
.tip{
  font-size: 15px;
  margin: 5px;
}
.right{
  margin-right: 5px;
}
.content{
  background-color: #F4F4F4;
  height: 600px;
}
.hr{
  height: 10px;
}
.item{
  background-color: #ffffff;
  width: 90%;
}

```



```

margin: 0 auto;
padding: 10px;
display: flex;
flex-direction: row;
align-items: center;
)
.station{
  font-size: 15px;
  font-weight: bold;
  margin-bottom: 5px;
}
.desc1{
  font-size: 15px;
  color: #999999;
  margin-bottom: 15px;
}
.desc2{
  font-size: 13px;
  color: #999999;
}
.ticket{
  width: 85%;
}
.opr{
  width: 45px;
  height: 45px;
  border-radius: 50px;
  background-color: #29CE73;
  color: #ffffff;
  line-height: 45px;
  font-size: 13px;
  text-align: center;
}
.start{
  width: 45px;
  height: 45px;
  border-radius: 50px;
  border: 1px solid red;
  color: red;
  line-height: 45px;
  font-size: 13px;
  text-align: center;
}

```

界面效果如图6.36所示。



图6.36 抢票信息

3. 分享信息设计

1 进入pages/ grabticket /grabticket.wxml文件，设计分享区域的内容，包括“加速”按钮、分享赢加速包以及进入二级界面的入口，具体代码如下。

```
<view class="amountBg">
  <view class="bg">
    <view class="icon"> 攻略 </view>
    <view class="tip"> 抢票成功后以电话或短信的方式通知，收到后请及时支付！ </view>
    <view class="right">></view>
  </view>
</view>
<view class="content">
<view class="hr"></view>
  <view class="item">
    <view class="ticket">
      <view class="station"> 哈尔滨→北京 </view>
      <view class="desc1">02月03日 </view>
      <view class="desc2">已取消 </view>
    </view>
    <view class="opr">
      查看
    </view>
  </view>
</view>
```

```

    </view>
  </view>
  <view class="hr"></view>
  <view class="item">
    <view class="ticket">
      <view class="station"> 北京→佳木斯 </view>
      <view class="desc1">02 月 07 日 </view>
      <view class="desc2"> 已取消 </view>
    </view>
    <view class="opr">
      查看
    </view>
  </view>
  <view class="hr"></view>
  <view class="item">
    <view class="ticket">
      <view class="station"> 北京→南京 </view>
      <view class="desc1">02 月 06 日 </view>
      <view class="desc2"> 已取消 </view>
    </view>
    <view class="start">
      开始
    </view>
  </view>
  <view class="hr"></view>
  <view class="item">
    <view class="ticket">
      <view class="station"> 合肥→北京 </view>
      <view class="desc1">02 月 07 日 </view>
      <view class="desc2"> 抢票终止 </view>
    </view>
    <view class="start">
      开始
    </view>
  </view>
</view>
<view class="share">
  <view class="speed"> 加速 </view>
  <view class="tipShare">
    <view> 分享赢加速包 </view>
    <view class="jsb"> 分享给好友，可随机赢取最多 10 个加速包 </view>
  </view>
  <view class="detail">></view>
</view>

```

2 进入pages/ grabticket /grabticket.wxss文件，给分享区域内容添加样式，设计“加速”按钮，具体代码如下。

```

.amountBg{
  height:70px;
  background-color: #5495E6;

```

```
}
.bg{
  width: 80%;
  height: 53px;
  background-color: #ffffff;
  border-radius: 5px;
  margin: 0 auto;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.icon{
  border: 1px solid #6EBAFE;
  font-size: 13px;
  margin-left: 10px;
  color: #6EBAFE;
  text-align: center;
}
.tip{
  font-size: 15px;
  margin: 5px;
}
.right{
  margin-right: 5px;
}
.content{
  background-color: #F4F4F4;
  height: 600px;
}
.hr{
  height: 10px;
}
.item{
  background-color: #ffffff;
  width: 90%;
  margin: 0 auto;
  padding: 10px;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.station{
  font-size: 15px;
  font-weight: bold;
  margin-bottom: 5px;
}
.descl{
  font-size: 15px;
  color: #999999;
  margin-bottom: 15px;
}
```



```

)
.desc2{
  font-size:13px;
  color: #999999;
}
.ticket{
  width: 85%;
}
.opr{
  width: 45px;
  height: 45px;
  border-radius: 50px;
  background-color: #29CE73;
  color: #ffffff;
  line-height: 45px;
  font-size: 13px;
  text-align: center;
}

.start{
  width: 45px;
  height: 45px;
  border-radius: 50px;
  border: 1px solid red;
  color: red;
  line-height: 45px;
  font-size: 13px;
  text-align: center;
}

.share{
  background-color: #ffffff;
  display: flex;
  flex-direction: row;
  position: fixed;
  bottom:0px;
  width: 100%;
  align-items: center;
  height: 55px;
}

.speed{
  width: 32px;
  height: 32px;
  border-radius: 50px;
  background-color: #29CE73;
  font-size: 11px;
  color: #ffffff;
  text-align: center;
  line-height: 32px;
  margin-left: 10px;
}

```

```
.tipShare{
  width: 80%;
}
.jsb{
  color: #999999;
  font-size: 13px;
  margin-top: 5px;
}
.detail{
  width: 15px;
  text-align: right;
}
```

界面效果如图6.37所示。



图6.37 分享区域

这样就完成了抢票界面的设计，通过界面的布局设计和样式设计，就可以实现抢票界面的显示。如果想改窗口标题，只需要在pages/ grabticket /grabticket.json文件里配置“navigationBarTitleText”：“抢票”属性，可以覆盖App.json里该属性。

6.4.7 项目上传和预览

仿智行火车票12306微信小程序后，如果该项目有AppID，可以将微信小程序上传到微信服务器上，扫描二维码并获得管理员的同意后即可上传项目，如图6.38所示。



图6.38 项目上传

除了项目上传，也可以进行项目预览。在手机上运行微信小程序，同样需要扫描二维码才能进行项目预览，如图6.39所示。



图6.39 项目预览

6.5

小结

本章主要设计了仿智行火车票12306微信小程序，重点掌握以下内容。

1 掌握底部标签导航的配置、顶部页签切换效果的设计，通过不同页签之间的切换，可以给用户展示动态的内容，实现不同内容的展示。

2 掌握表单组件提交表单内容，以及将这些表单内容传递到其他界面、其他界面获取传递的内容。

3 掌握界面的布局以及给界面布局添加相应的样式，添加相应的绑定事件。

4 学会将js里的数据动态地绑定到WXML界面，实现数据的动态绑定。

5 学会设计列表内容或者列表导航界面，通过先设计一个基本的内容区域，然后复用这个区域的内容，来提高开发效率。

6 学会wx.request、wx.navigateTo、wx.setNavigationBarTitle等开发接口的使用，每个API有不同的用处，达到不同的效果。



图7.5 购票界面

扫码求索



第7章 综合案例：仿糗事百科微信小程序

糗事百科App是以糗友的真实糗事为主题的笑话App，话题轻松休闲，在年轻人中十分流行。在糗事百科中可以查看他人发布的糗事并与网友分享自己亲身经历或听说到的各类情形的生活糗事，如图7.1和图7.2所示。

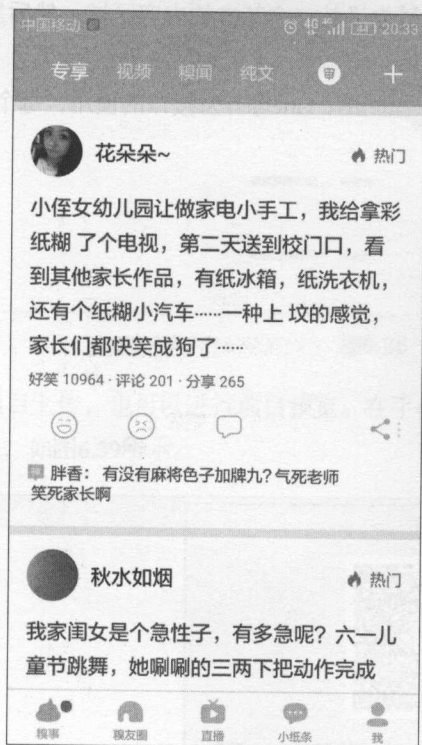


图7.1 专享



图7.2 视频

7.1 需求描述

仿糗事百科微信小程序主要完成以下功能。

1 实现顶部页签菜单左右滑动的效果，如图7.3所示。

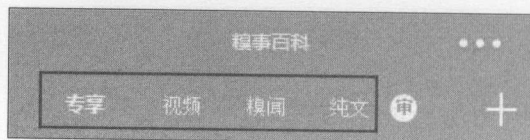


图7.3 顶部页签

2 实现顶部页签菜单的切换效果，页签菜单选中时字体加粗，同时对应的内容也跟着变化，如图7.4和图7.5所示。

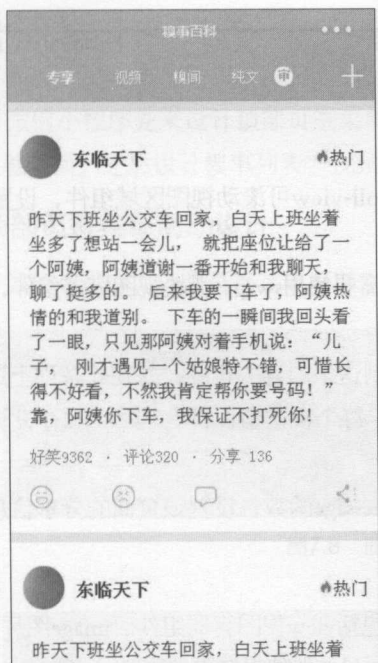


图7.4 专享界面



图7.5 视频界面

3 实现专享界面糗事列表的设计，包括发布人头像、发布人昵称、发布的段子等信息，以列表的形式展现出来。

4 实现视频列表页的设计，使视频可以进行播放与暂停。

5 实现分享功能，可以将当前界面分享给好友，如图7.6所示。

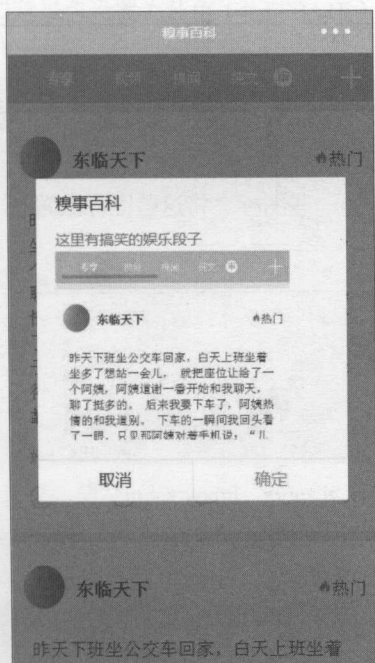


图7.6 分享页面

7.2 设计思路及相关知识点

7.2.1 设计思路

1 实现顶部页签的滑动效果。需要借助于scroll-view可滚动视图区域组件，设置scroll-x="true"属性，允许在水平方向上左右滑动。

2 页签菜单切换，其内容也跟着进行切换。需要使用swiper滑块视图容器组件，根据current当前页面索引值来决定显示的面板。

3 设计糗事列表。先设计一条内容，然后复制这条内容的布局，并在此基础上进行修改。

4 设计视频列表。需要使用video视频组件，每个视频组件都有唯一的id；设计幻灯片轮播效果，准备好幻灯片需要轮播的图片。

5 分享功能。需要在Page中定义onShareAppMessage函数，设置该页面的分享信息。

7.2.2 相关知识点

1 在界面布局时会用到微信小程序的组件，包括view视图容器组件、image图片组件、swiper滑块视图容器组件、scroll-view可滚动视图区域组件、video视频组件等。

2 在设计界面样式时，需要写一些wxss样式进行界面的美化和渲染。

3 在进行页签菜单切换时，需要获得该页签所对应的id，绑定菜单切换事件。

4 进行页面分享，需要使用onShareAppMessage这个API接口。

5 动态地获取糗事列表信息，需要使用wx.request请求。

7.3 准备工作

1 需要准备一个AppID，如果没有AppID也没有关系，只不过不能再在手机上进行项目的预览了，但是在开发工具上开发是没有任何问题的。

2 在设计列表页时，需要用到一些图标，将这些图标放置在“images/icon”文件夹下，如图7.7所示。

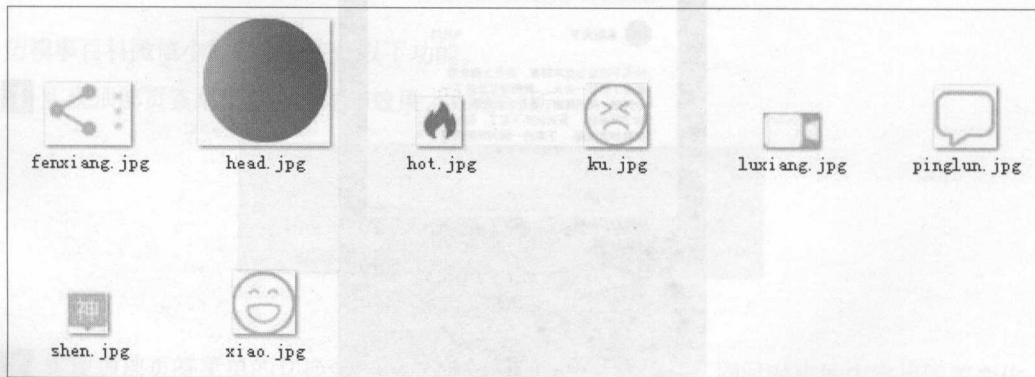


图7.7 图标

7.4 设计流程

仿糗事百科微信小程序先来设计顶部页签菜单左右滑动效果、页签菜单切换效果，切换时页签对应的内容也跟着切换，之后设计糗事列表、视频列表，最后设计分享功能和项目预览。

7.4.1 顶部页签菜单滑动设计

仿糗事百科微信小程序，顶部页签菜单可以左右滑动，如图7.8和图7.9所示。

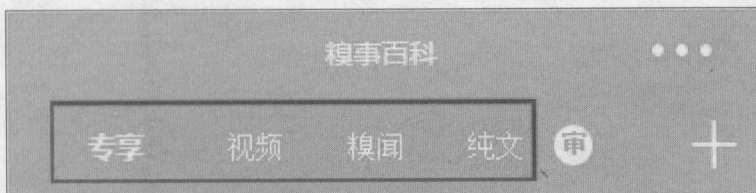


图7.8 顶部菜单一

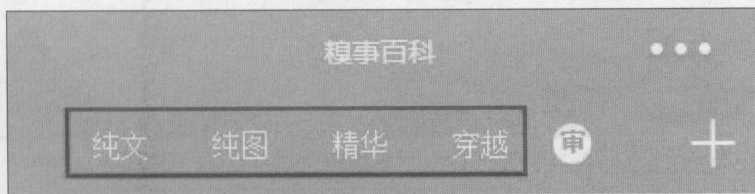


图7.9 顶部菜单二

1 新建一个qsbk项目，AppID为wxa7730e0596be9404，把准备好的图片放置在qsbk项目里。

2 进入App.json界面，将窗口背景色设置为黄色（#FFBA1E），标题改为“糗事百科”，字体颜色设置为白色（white），具体代码如下。

```
{
  "pages": [
    "pages/index/index",
    "pages/logs/logs"
  ],
  "window": {
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#FFBA1E",
    "navigationBarTitleText": "糗事百科",
    "navigationBarTextStyle": "white"
  }
}
```

3 进入到pages/index/index目录下，清空index.wxml、index.js、index.wxss文件默认生成的内容。

4 进入pages/index/index.wxml文件，设计顶部页签菜单，包括页签菜单、审字、+，具体代码如下。

```
<view class="bg">
  <view class="nav">
    <scroll-view class="scroll-view_H" scroll-x="true">
```

```

<view class="scroll-view_H">
  <view><view class="{{flag==0?'select':'normal'}}" id="0" bindtap="switchNav">
专享 </view></view>
  <view><view class="{{flag==1?'select':'normal'}}" id="1" bindtap="switchNav">
视频 </view></view>
  <view><view class="{{flag==2?'select':'normal'}}" id="2" bindtap="switchNav">
糗闻 </view></view>
  <view><view class="{{flag==3?'select':'normal'}}" id="3" bindtap="switchNav">
纯文 </view></view>
  <view><view class="{{flag==4?'select':'normal'}}" id="4" bindtap="switchNav">
纯图 </view></view>
  <view><view class="{{flag==5?'select':'normal'}}" id="5" bindtap="switchNav">
精华 </view></view>
  <view><view class="{{flag==6?'select':'normal'}}" id="6" bindtap="switchNav">
穿越 </view></view>
</view>
</scroll-view>
</view>
<view class="opr">
  申
</view>
<view class="add">+</view>
</view>

```

5 进入pages/index/index.wxss文件，给顶部菜单内容添加样式，具体代码如下。

```

.bg{
  background-color: #FFBA1E;
  height: 50px;
  color: #ffffff;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.nav{
  width: 70%;
  height: 40px;
}
.opr{
  width: 20px;
  height: 20px;
  border-radius: 50%;
  font-size: 13px;
  line-height: 20px;
  text-align: center;
  color: #FFBA1E;
  background-color: #ffffff;
  font-weight: bold;
  margin-left: 10px;
}
.add{
  width: 20%;

```



```
height: 50px;
line-height: 50px;
text-align: right;
margin-right: 10px;
font-size: 50px;
}
.scroll-view_H{
margin-left: 10px;
height: 40px;
display: flex;
flex-direction: row;
}
.normal{
width: 40px;
height: 40px;
line-height: 40px;
padding-left: 10px;
padding-right: 10px;
font-size: 14px;
}
.select{
width: 40px;
height: 40px;
line-height: 40px;
padding-left: 20px;
padding-right: 20px;
font-size: 14px;
font-weight: bold;
}
```

这样就可以实现顶部页签左右滑动效果，如图7.10所示。

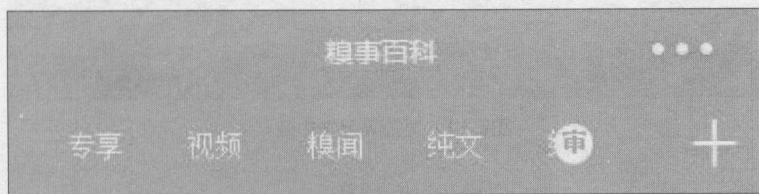


图7.10 顶部菜单内容

7.4.2 顶部页签菜单切换效果设计

顶部页签菜单可以左右切换，但还没有实现页签菜单的切换效果，单击不同的页签，页签需要呈现为选中状态，同时页签对应的内容也跟着切换。

1 在页签里设计两种样式：一种是select样式，选中时字体加粗；另一种是normal样式，字体不加粗。绑定单击事件switchNav。

2 进入pages/index/index.js文件，定义两个变量：currentTab当前页签的索引值，flag变量用来控制样式选择，如果flag等于页签对应的id，则呈现为选中状态，使用select样式，具体代码如下。

```
Page({
  data:{
    currentTab:0,
    flag:0
  }
})
```

3 添加页签菜单单击绑定事件switchNav，动态地给currentTab和flag变量赋值，具体代码如下。

```
Page({
  data:{
    currentTab:0,
    flag:0
  },
  switchNav:function(e){
    console.log(e);
    var page = this;
    var id = e.target.id;
    if(this.data.currentTab == id){
      return false;
    }else{
      page.setData({currentTab:id});
    }
    page.setData({flag:id});
  }
})
```

这样单击不同页签，页签会呈现出选中效果，如图7.11所示。

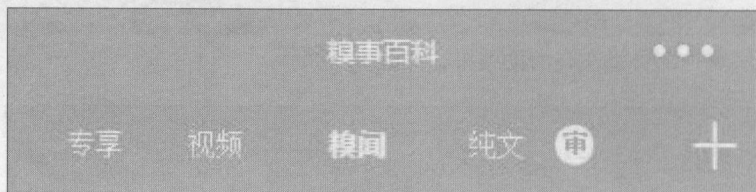


图7.11 页签菜单选中效果

4 页签菜单进行切换时，对应的内容也跟着切换，需要使用swiper滑块视图容器组件，进入到pages/index/index.wxml文件里，使用swiper进行页签内容的布局，具体代码如下。

```
<view class="bg">
  <view class="nav">
    <scroll-view class="scroll-view_H" scroll-x="true" >
      <view class="scroll-view_H">
        <view><view class="{{flag==0?'select':'normal'}}" id="0" bindtap="switchNav">
专享 </view></view>
        <view><view class="{{flag==1?'select':'normal'}}" id="1" bindtap="switchNav">
视频 </view></view>
        <view><view class="{{flag==2?'select':'normal'}}" id="2" bindtap="switchNav">
糗闻 </view></view>
        <view><view class="{{flag==3?'select':'normal'}}" id="3" bindtap="switchNav">
```

```

纯文 </view></view>
      <view><view class="{{flag==4?'select':'normal'}}" id="4" bindtap="switchNav">
纯图 </view></view>
      <view><view class="{{flag==5?'select':'normal'}}" id="5" bindtap="switchNav">
精华 </view></view>
      <view><view class="{{flag==6?'select':'normal'}}" id="6" bindtap="switchNav">
穿越 </view></view>
    </view>
  </scroll-view>
</view>
  <view class="opr">
    审
  </view>
  <view class="add">+</view>
</view>
<swiper current="{{currentTab}}" style="height:1500px">
  <swiper-item>
    我是专享内容
  </swiper-item>
  <swiper-item>
    我是视频内容
  </swiper-item>
  <swiper-item>
    我是模闻内容
  </swiper-item>
  <swiper-item>
    我是纯文内容
  </swiper-item>
  <swiper-item>
    我是纯图内容
  </swiper-item>
  <swiper-item>
    我是精华内容
  </swiper-item>
  <swiper-item>
    我是穿越内容
  </swiper-item>
</swiper>

```

这样单击页签菜单时，不仅菜单标题呈现为选中状态，同时页签内容也跟着变化，实现页签菜单和页签内容的联动效果。

7.4.3 糗事列表页设计

糗事列表页面是用来显示糗事内容的页面，每条糗事包括4方面的内容：①发布人头像、发布人昵称、热门图标；②发布的糗事内容；③糗事好笑数量、评论数量、分享数量；④好笑、评论、分享的操作按钮图标，如图7.12所示。

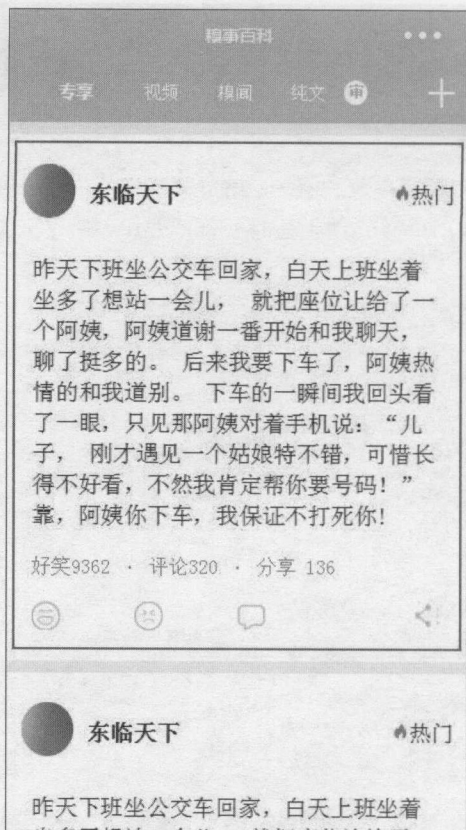


图7.12 糗事列表内容

1 在pages/index目录下新建一个vip.wxxml文件, 然后进入pages/index/index.wxxml文件, 引入vip.wxxml文件, 具体代码如下。

```
<view class="bg">
  <view class="nav">
    <scroll-view class="scroll-view_H" scroll-x="true" >
      <view class="scroll-view_H">
        <view><view class="{{flag==0?'select':'normal'}}" id="0" bindtap="switchNav">
专享</view></view>
        <view><view class="{{flag==1?'select':'normal'}}" id="1" bindtap="switchNav">
视频</view></view>
        <view><view class="{{flag==2?'select':'normal'}}" id="2" bindtap="switchNav">
糗闻</view></view>
        <view><view class="{{flag==3?'select':'normal'}}" id="3" bindtap="switchNav">
纯文</view></view>
        <view><view class="{{flag==4?'select':'normal'}}" id="4" bindtap="switchNav">
纯图</view></view>
        <view><view class="{{flag==5?'select':'normal'}}" id="5" bindtap="switchNav">
精华</view></view>
        <view><view class="{{flag==6?'select':'normal'}}" id="6" bindtap="switchNav">
穿越</view></view>
      </view>
    </scroll-view>
  </view>
</view>
```

```

</view>
<view class="opr">
  审
</view>
<view class="add">+</view>
</view>
<swiper current="{{currentTab}}" style="height:1500px">
  <swiper-item>
    <include src="vip.wxml"/>
  </swiper-item>
  <swiper-item>
    我是视频内容
  </swiper-item>
  <swiper-item>
    我是糗闻内容
  </swiper-item>
  <swiper-item>
    我是纯文内容
  </swiper-item>
  <swiper-item>
    我是纯图内容
  </swiper-item>
  <swiper-item>
    我是精华内容
  </swiper-item>
  <swiper-item>
    我是穿越内容
  </swiper-item>
</swiper>

```

2 进入pages/index/vip.wxml文件，设计发布人头像、发布人昵称、热门图标，具体代码如下。

```

<view class="line"></view>
<view class="item">
  <view class="head">
    <view><image src="../../images/icon/head.jpg" style="width:45px;height:45px;"></image></view>
    <view class="title">东临天下</view>
    <view class="hot"><image src="../../images/icon/hot.jpg" style="width:14px;height:16px;"></image> 热门</view>
  </view>
</view>

```

3 进入pages/index/index.wxss文件，给发布人头像、发布人昵称、热门图标添加样式，具体代码如下。

```

.bg{
  background-color: #FFBA1E;
  height: 50px;
  color: #ffffff;
  display: flex;

```

```

    flex-direction: row;
    align-items: center;
  }
  .nav{
    width: 70%;
    height: 40px;
  }
  .opr{
    width: 20px;
    height: 20px;
    border-radius:50%;
    font-size: 13px;
    line-height: 20px;
    text-align: center;
    color: #FFBA1E;
    background-color: #ffffff;
    font-weight: bold;
    margin-left: 10px;
  }
  .add{
    width: 20%;
    height: 50px;
    line-height: 50px;
    text-align: right;
    margin-right:10px;
    font-size: 50px;
  }
  .scroll-view_H{
    margin-left: 10px;
    height: 40px;
    display: flex;
    flex-direction: row;
  }
  .normal{
    width: 40px;
    height: 40px;
    line-height: 40px;
    padding-left:10px;
    padding-right: 10px;
    font-size: 14px;
  }
  .select{
    width: 40px;
    height: 40px;
    line-height: 40px;
    padding-left:20px;
    padding-right: 20px;
    font-size: 14px;
    font-weight: bold;
  }
}

```



```

.line{
  height: 10px;
  background-color: #F2F2F2;
}
.item{
  margin: 10px;
}
.head{
  display: flex;
  flex-direction: row;
  height: 77px;
  align-items: center;
}
.title{
  width: 60%;
  margin-left: 10px;
  font-weight: bold;
}
.hot{
  text-align: right;
  width: 30%;
}

```

界面效果如图7.13所示。

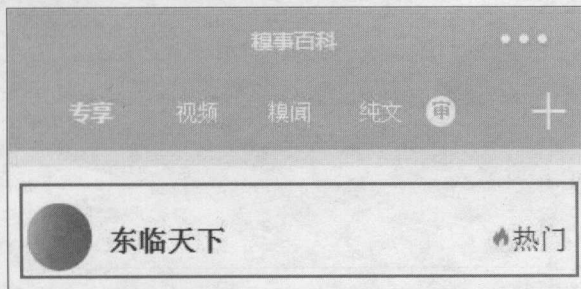


图7.13 发布人头像和昵称

4 进入pages/index/vip.wxml文件，设计糗事内容、好笑数量、评论数量、分析数量，具体代码如下。

```

<view class="line"></view>
<view class="item">
  <view class="head">
    <view><image src="../../images/icon/head.jpg" style="width:45px;height:45px;"></image></view>
    <view class="title">东临天下</view>
    <view class="hot"><image src="../../images/icon/hot.jpg" style="width:14px;height:16px;"></image>热门</view>
  </view>

  <view class="content">
    昨天下班坐公交车回家，白天上班坐着坐多了想站一会儿，\r\n就把座位让给了一个阿姨，阿姨道谢一番开

```

始和我聊天，聊了挺多的。\\r\\n 后来我要下车了，阿姨热情的和我道别。\\r\\n 下车的一瞬间我回头看了一眼，只见那阿姨对着手机说：“儿子，\\r\\n 刚才遇见一个姑娘特不错，可惜长得不好看，不然我肯定帮你要号码！”\\r\\n 靠，阿姨你下车，我保证不打死你！

```
</view>
<view class="sta">好笑 9362 • 评论 320 • 分享 136</view>
</view>
```

5 进入pages/index/index.wxss文件，给糗事内容、好笑数量、评论数量、分析数量添加样式，具体代码如下。

```
.bg{
  background-color: #FFBA1E;
  height: 50px;
  color: #ffffff;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.nav{
  width: 70%;
  height: 40px;
}
.opr{
  width: 20px;
  height: 20px;
  border-radius: 50%;
  font-size: 13px;
  line-height: 20px;
  text-align: center;
  color: #FFBA1E;
  background-color: #ffffff;
  font-weight: bold;
  margin-left: 10px;
}
.add{
  width: 20%;
  height: 50px;
  line-height: 50px;
  text-align: right;
  margin-right: 10px;
  font-size: 50px;
}
.scroll-view_H{
  margin-left: 10px;
  height: 40px;
  display: flex;
  flex-direction: row;
}
.normal{
  width: 40px;
  height: 40px;
```

```

    line-height: 40px;
    padding-left: 10px;
    padding-right: 10px;
    font-size: 14px;
  }
  .select{
    width: 40px;
    height: 40px;
    line-height: 40px;
    padding-left: 20px;
    padding-right: 20px;
    font-size: 14px;
    font-weight: bold;
  }
  .line{
    height: 10px;
    background-color: #F2F2F2;
  }
  .item{
    margin: 10px;
  }
  .head{
    display: flex;
    flex-direction: row;
    height: 77px;
    align-items: center;
  }
  .title{
    width: 60%;
    margin-left: 10px;
    font-weight: bold;
  }
  .hot{
    text-align: right;
    width: 30%;
  }
  .content{
    padding: 10px;
    line-height: 25px;
  }
  .sta{
    padding: 10px;
    color: #999999;
    font-size: 16px;
  }
}

```

6 进入pages/index/vip.wxml文件，设计好笑、不好笑、评论、分享图标，具体代码如下。

```

<view class="line"></view>
<view class="item">
  <view class="head">
    <view><image src="../../../images/icon/head.jpg" style="width:45px;height:45px;">

```



```

</image></view>
  <view class="title">东临天下</view>
  <view class="hot"><image src="../../../images/icon/hot.jpg" style="width:14px;height:16px;"></image> 热门</view>
</view>

<view class="content">
  昨天下班坐公交车回家，白天上班坐着坐多了想站一会儿，\r\n就把座位让给了一个阿姨，阿姨道谢一番开始和我聊天，聊了挺多的。\r\n后来我要下车了，阿姨热情的和我道别。\r\n下车的一瞬间我回头看了一眼，只见那阿姨对着手机说："儿子，\r\n刚才遇见一个姑娘特不错，可惜长得不好看，不然我肯定帮你要号码！"\r\n靠，阿姨你下车，我保证不打死你！
</view>
  <view class="sta">好笑 9362 • 评论 320 • 分享 136</view>
  <view class="icon">
    <view><image src="../../../images/icon/xiao.jpg" style="width:23px;height:23px;"></image></view>
    <view><image src="../../../images/icon/ku.jpg" style="width:23px;height:23px;"></image></view>
    <view><image src="../../../images/icon/pinglun.jpg" style="width:23px;height:23px;"></image></view>
    <view class="last"><image src="../../../images/icon/fenxiang.jpg" style="width:23px;height:23px;"></image></view>
  </view>
</view>

```

7 进入pages/index/index.wxss文件，给好笑、不好笑、评论、分享图标添加样式，具体代码如下。

```

.bg{
  background-color: #FFBA1E;
  height: 50px;
  color: #ffffff;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.nav{
  width: 70%;
  height: 40px;
}
.opr{
  width: 20px;
  height: 20px;
  border-radius:50%;
  font-size: 13px;
  line-height: 20px;
  text-align: center;
  color: #FFBA1E;
  background-color: #ffffff;
  font-weight: bold;
  margin-left: 10px;
}

```

```
}  
.add{  
  width: 20%;  
  height: 50px;  
  line-height: 50px;  
  text-align: right;  
  margin-right: 10px;  
  font-size: 50px;  
}  
.scroll-view_H{  
  margin-left: 10px;  
  height: 40px;  
  display: flex;  
  flex-direction: row;  
}  
.normal{  
  width: 40px;  
  height: 40px;  
  line-height: 40px;  
  padding-left: 10px;  
  padding-right: 10px;  
  font-size: 14px;  
}  
.select{  
  width: 40px;  
  height: 40px;  
  line-height: 40px;  
  padding-left: 20px;  
  padding-right: 20px;  
  font-size: 14px;  
  font-weight: bold;  
}  
.line{  
  height: 10px;  
  background-color: #F2F2F2;  
}  
.item{  
  margin: 10px;  
}  
.head{  
  display: flex;  
  flex-direction: row;  
  height: 77px;  
  align-items: center;  
}  
.title{  
  width: 60%;  
  margin-left: 10px;  
  font-weight: bold;  
}
```

```
.hot{
  text-align: right;
  width: 30%;
}
.content{
  padding: 10px;
  line-height: 25px;
}
.sta{
  padding: 10px;
  color: #999999;
  font-size: 16px;
}
.icon{
  padding: 10px;
  display: flex;
  flex-direction: row;
}
.icon view{
  margin: 0 auto;
  width: 25%;
}
.last{
  text-align: right;
}
.section{
  text-align: center;
}
```

界面效果如图7.14所示。

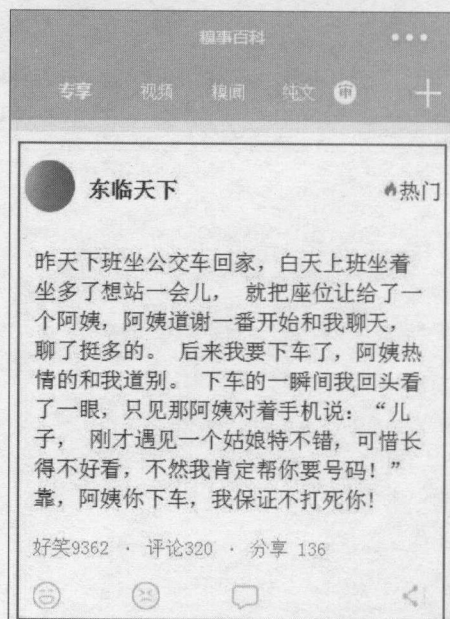


图7.14 糗事内容

8 把这块区域作为列表的基础单元内容，循环展现出列表，只需要修改发布人头像、发布人昵称、糗事以及好笑评论分享数量，其余界面布局和样式不需要修改。

7.4.4 视频列表页设计

视频列表页用来放置一些搞笑视频或者随拍视频，需要使用video视频组件，其界面布局方式和糗事列表布局方式一样，如图7.15所示。



图7.15 视频列表

1 在pages/index目录下新建一个video.wxml页面，然后将这个页面引入到index.wxml文件里，具体代码如下。

```
<view class="bg">
  <view class="nav">
    <scroll-view class="scroll-view_H" scroll-x="true">
      <view class="scroll-view_H">
        <view><view class="{{flag==0?'select':'normal'}}" id="0" bindtap="switchNav">
专享</view></view>
        <view><view class="{{flag==1?'select':'normal'}}" id="1" bindtap="switchNav">
视频</view></view>
        <view><view class="{{flag==2?'select':'normal'}}" id="2" bindtap="switchNav">
糗闻</view></view>
```

```

        <view><view class="{{flag==3?'select':'normal'}}" id="3" bindtap="switchNav">
纯文 </view></view>
        <view><view class="{{flag==4?'select':'normal'}}" id="4" bindtap="switchNav">
纯图 </view></view>
        <view><view class="{{flag==5?'select':'normal'}}" id="5" bindtap="switchNav">
精华 </view></view>
        <view><view class="{{flag==6?'select':'normal'}}" id="6" bindtap="switchNav">
穿越 </view></view>
    </view>
</scroll-view>
</view>
<view class="opr">
    审
</view>
<view class="add">+</view>
</view>
<swiper current="{{currentTab}}" style="height:1500px">
    <swiper-item>
        <include src="vip.wxml"/>
    </swiper-item>
    <swiper-item>
        <include src="video.wxml"/>
    </swiper-item>
    <swiper-item>
        我是模闻内容
    </swiper-item>
    <swiper-item>
        我是纯文内容
    </swiper-item>
    <swiper-item>
        我是纯图内容
    </swiper-item>
    <swiper-item>
        我是精华内容
    </swiper-item>
    <swiper-item>
        我是穿越内容
    </swiper-item>
</swiper>

```

2 将vip.wxml文件内容拷贝到video.wxml文件里，在这个基础上修改video.wxml文件，将其改成视频列表页面，具体代码如下。

```

<view class="line"></view>
<view class="item">
    <view class="head">
        <view><image src="../../images/icon/head.jpg" style="width:45px;height:45px;">
</image></view>
        <view class="title">东临天下 </view>
    </view>

```

```

<view class="section">
  <video style="width:320px" id="myVideo" src="http://wxsnsdy.tc.qq.com/105/20210/sns
dyvideodownload?filekey=30280201010421301f0201690402534804102ca905ce620b1241b726bc41dc
ff44e00204012882540400&bizid=1023&hy=SH&fileparam=302c020101042530230204136ffd93020457
e3c4ff02024ef202031e8d7f02030f42400204045a320a0201000400" controls></video>
</view>
<view class="sta">好笑 9362 · 评论 320 · 分享 136</view>
  <view class="icon">
    <view><image src="../../images/icon/xiao.jpg" style="width:23px;height:23px;"></
image></view>
    <view><image src="../../images/icon/ku.jpg" style="width:23px;height:23px;"></
image></view>
    <view><image src="../../images/icon/pinglun.jpg" style="width:23px;height:23
px;"></image></view>
    <view class="last"><image src="../../images/icon/fenxiang.jpg" style="width:23
px;height:23px;"></image></view>
  </view>
</view>

<view class="line"></view>
<view class="item">
  <view class="head">
    <view><image src="../../images/icon/head.jpg" style="width:45px;height:45px;"></
image></view>
    <view class="title">人生有太多的遗憾</view>
  </view>

  <view class="section">
    <video style="width:320px" id="myVideo1" src="http://wxsnsdy.tc.qq.com/105/20210/sn
sdyvideodownload?filekey=30280201010421301f0201690402534804102ca905ce620b1241b726bc41d
cfff44e00204012882540400&bizid=1023&hy=SH&fileparam=302c020101042530230204136ffd9302045
7e3c4ff02024ef202031e8d7f02030f42400204045a320a0201000400" controls></video>
    </view>
    <view class="sta">好笑 9362 · 评论 320 · 分享 136</view>
    <view class="icon">
      <view><image src="../../images/icon/xiao.jpg" style="width:23px;height:23px;"></
image></view>
      <view><image src="../../images/icon/ku.jpg" style="width:23px;height:23px;"></
image></view>
      <view><image src="../../images/icon/pinglun.jpg" style="width:23px;height:23
px;"></image></view>
      <view class="last"><image src="../../images/icon/fenxiang.jpg" style="width:23
px;height:23px;"></image></view>
    </view>
  </view>
</view>

```

3 video界面布局样式不需要修改，可以共用index.wxss样式，界面效果如图7.16所示。



图7.16 视频内容

7.4.5 分享设计

微信小程序支持分享页功能，可以将当前页分享给好友，好友看到这个页面时页面显示的是实时数据，可以直接进入微信小程序，单击右上角就可以进行分享，如图7.17所示。

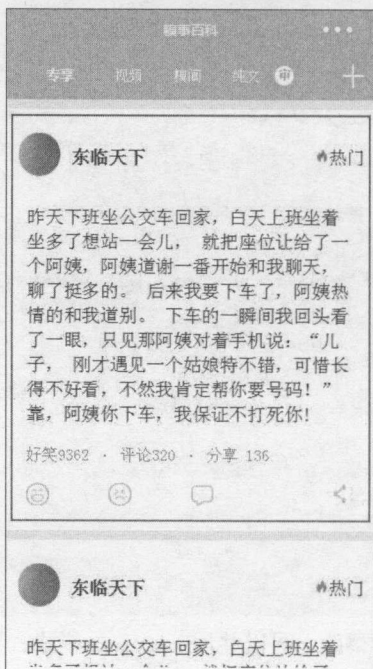


图7.17 分享设计

进入pages/index/index.js文件，添加onShareAppMessage分享功能接口函数，具体代码如下。

```
Page({
  data: {
    currentTab: 0,
    flag: 0
  },
  switchNav: function (e) {
    console.log(e);
    var page = this;
    var id = e.target.id;
    if (this.data.currentTab == id) {
      return false;
    } else {
      page.setData({ currentTab: id });
    }
    page.setData({ flag: id });
  },
  onShareAppMessage: function () {
    return {
      title: '糗事百科',
      desc: '这里有搞笑的娱乐段子',
      path: '/index/index'
    }
  }
})
```

在微信开发者工具上单击分享功能时，界面效果如图7.18所示。

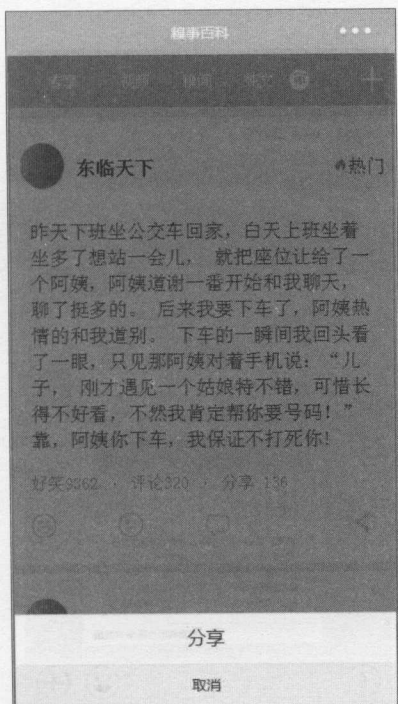


图7.18 分享功能

7.4.6 项目预览

项目有AppID的，可以将项目进行上传，在手机上可以浏览仿糗事百科微信小程序，如图7.19~图7.21所示。

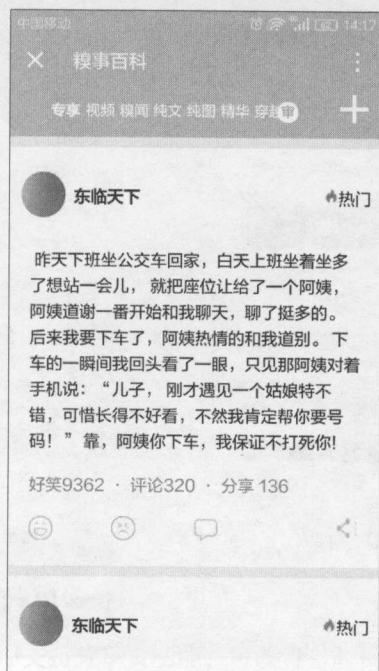


图7.19 糗事列表

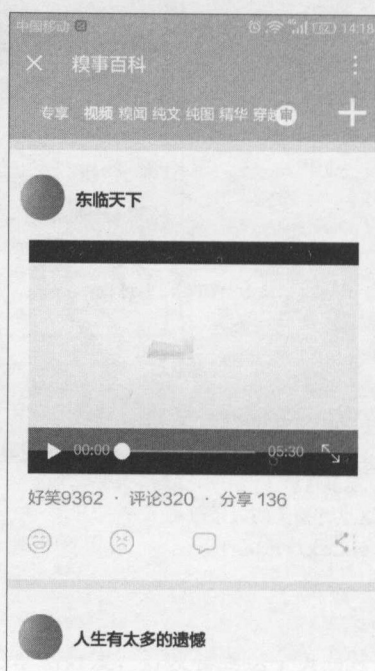


图7.20 视频列表

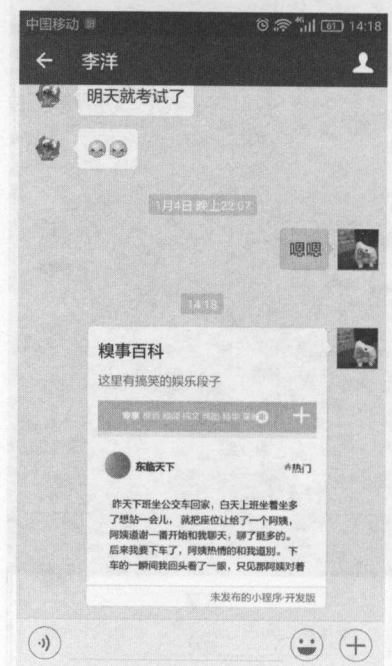


图7.21 分享

7.5 小结

本章主要设计了仿糗事百科微信小程序，重点掌握以下内容。

- 1 学会使用scroll-view可滚动视图组件来设计左右滑动效果。
- 2 学会设计页签菜单和页签内容联动切换效果。
- 3 学会设计糗事列表的布局以及使用引入文件，复用糗事列表内容，提高开发效率。
- 4 学会设计分享功能，借助于微信小程序提供的分享功能接口，将当前页面分享给好友。
- 5 学会界面布局和给界面添加相关的样式。



图 7-8-1 在首页显示分类图标、搞笑段子、糗事百科相关内容，底部有相关内容的引导



图 7-8-2 显示单个界面，底部有分享、收藏按钮

第8章 综合案例：仿中国婚博会微信小程序

中国婚博会每个季度举办一次，为准备结婚的新人们提供一条龙服务，包括婚纱摄影、酒店、珠宝首饰、婚庆、租婚车、租婚纱等。参加婚博会需要使用中国婚博会App索要门票、领签到礼。由于中国婚博会App软件的使用频率不是很高，所以完全可以做一个中国婚博会小程序，需要的时候直接搜索出来使用，主要界面如图8.1~图8.4所示。



图8.1 首页



图8.2 全部分类



图8.3 现金券

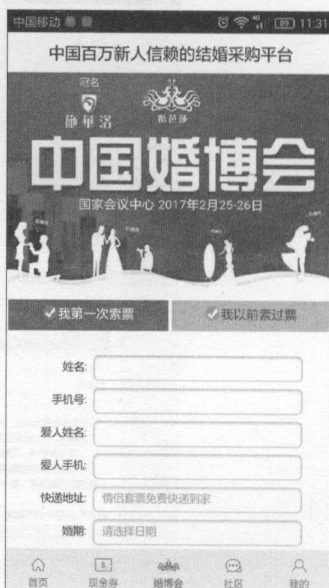


图8.4 婚博会

8.1 需求描述

仿中国婚博会微信小程序要实现以下主要功能。

1 完成底部标签导航的设计、首页海报轮播效果的设计和宫格导航的设计，如图8.5所示。



图8.5 首页设计

2 在首页里，单击全部分类宫格导航时，会进入全部分类导航界面，把婚博会相关内容的导航集成到一个界面里，如图8.6所示。



图8.6 全部分类界面

3 在现金券界面里，将各个商户的现金券以列表的形式展现出来，提供全部、默认下拉菜单效果显示，如图8.7所示。



图8.7 现金券界面

4 在婚博会界面里，提供索票的界面，填写个人相关信息后，可以进行索票，如图8.8所示。

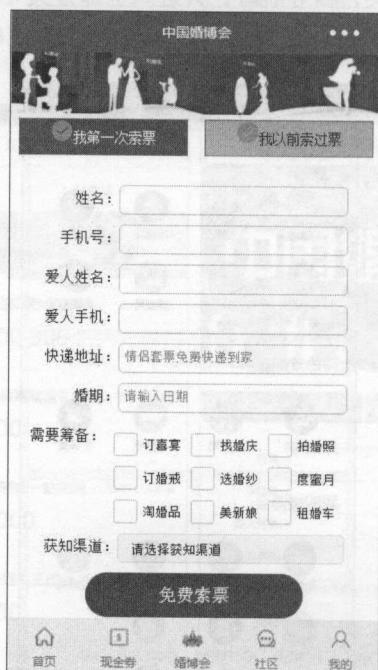


图8.8 免费索票

5 在填写表单选择获知渠道时，以弹出窗口的形式提供单选列表，供用户对获知渠道进行选择，如图8.9所示。

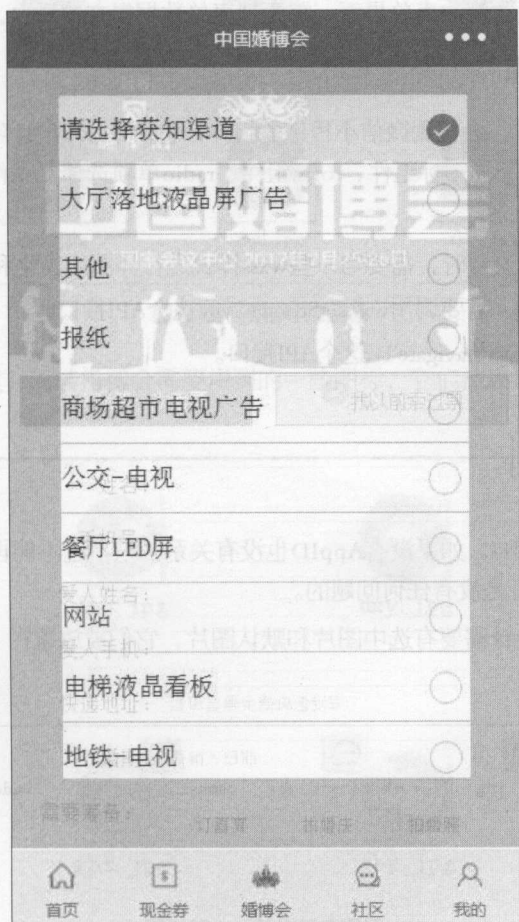


图8.9 选择获知渠道

8.2 设计思路及相关知识点

8.2.1 设计思路

1 设计底部标签导航，准备好底部标签导航的图标和建立相应的5个页面；设置默认时图片和选中时图片，标签名称采用两种颜色，红色为选中颜色，灰色为默认颜色。

2 设计幻灯片轮播效果，准备好幻灯片需要轮播的图片。

3 设计宫格导航时，先把宫格导航的图标和导航名称存放在js后台里，动态循环展现出宫格导航。

4 在设计全部分类导航时，有3块区域内容：玩转婚博会、特色分类、我的婚博会，由于这3个界面的布局方式一样，可以先设计出一个区域，剩下的两个直接复制使用。

5 现金券界面的设计难点在于下拉菜单筛选条件的设计，需要把筛选条件置于页面顶层，可以通过在样式里设置`z-index:999`来完成。

6 婚博会索票界面是常规的表单界面，需要把表单数据提交给后台，保存到本地。

8.2.2 相关知识点

1 在界面布局的时候，会用到微信小程序的组件，包括`view`视图容器组件、`image`图片组件、`swiper`滑块视图容器组件、`icon`图标组件、`form`组件、`radio`单项选择器组件、`checkbox`多项选择器组件等。

2 在进行界面样式的设计时，需要写一些`wxss`样式进行界面的美化和渲染。

3 将数据缓存到本地，需要调用`wx.setStorageSync`这个API接口。

4 界面跳转需要使用`wx.navigateTo`这个API接口。

8.3 准备工作

1 需要准备一个AppID，如果没有AppID也没有关系，只不过不能再在手机上进行项目的预览了，但是在开发工具上开发是没有任何问题的。

2 底部标签导航的设计需要有选中图片和默认图片，它们被放置在“`images/bar`”文件夹下，如图8.10所示。

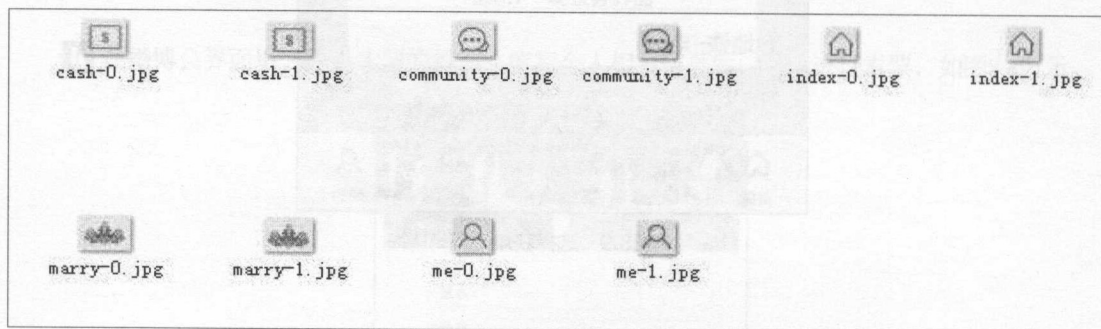


图8.10 底部标签导航图片

3 需要准备海报轮播的图片，它们被放置在“`images/haibao`”文件夹下，如图8.11所示。

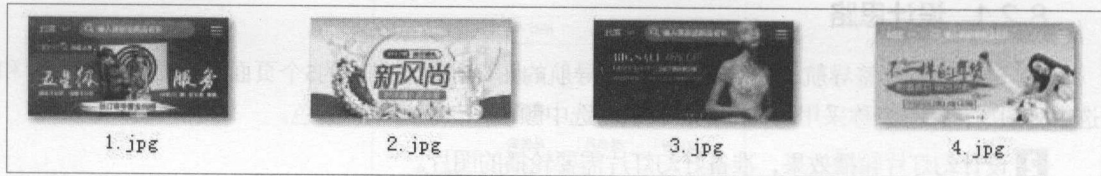


图8.11 海报轮播图片

4 在首页设计宫格导航时需要用到一些图标，它们被放置在“`images/nav`”文件夹下，如图8.12所示。

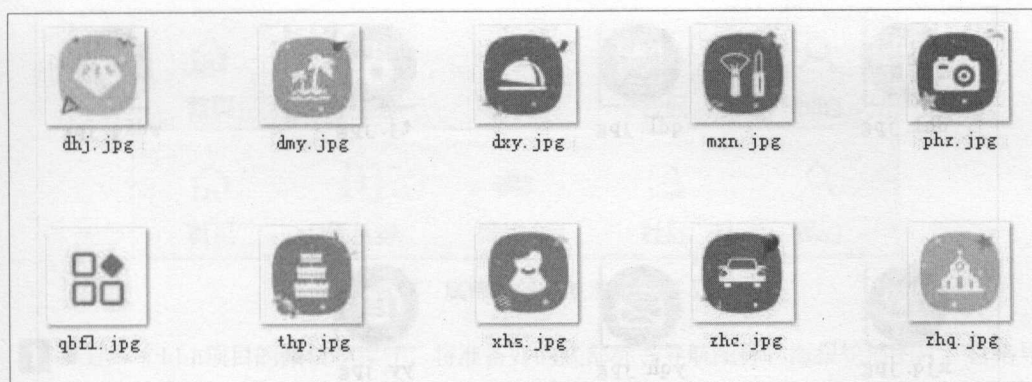


图8.12 宫格导航图标

5 在全部分类界面里，玩转婚博会需要用到的一些图标，它们被放置在“images/type/wzhbh”文件夹下，如图8.13所示。

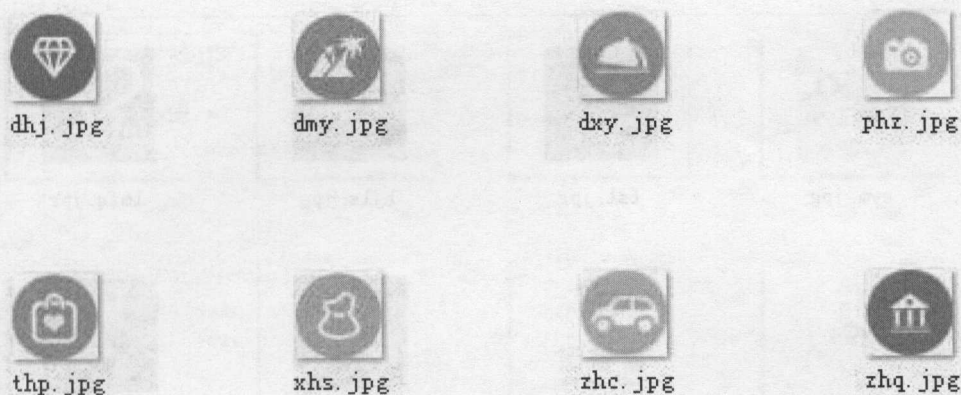


图8.13 玩转婚博会图标

6 在全部分类界面里，特色分类需要用到的一些图标，它们被放置在“images/type/tsfl”文件夹下，如图8.14所示。

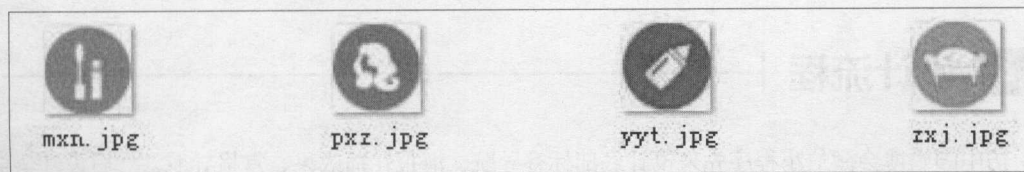


图8.14 特色分类图标

7 在全部分类界面里，我的婚博会需要用到的一些图标，它们被放置在“images/type/wdhbh”文件夹下，如图8.15所示。

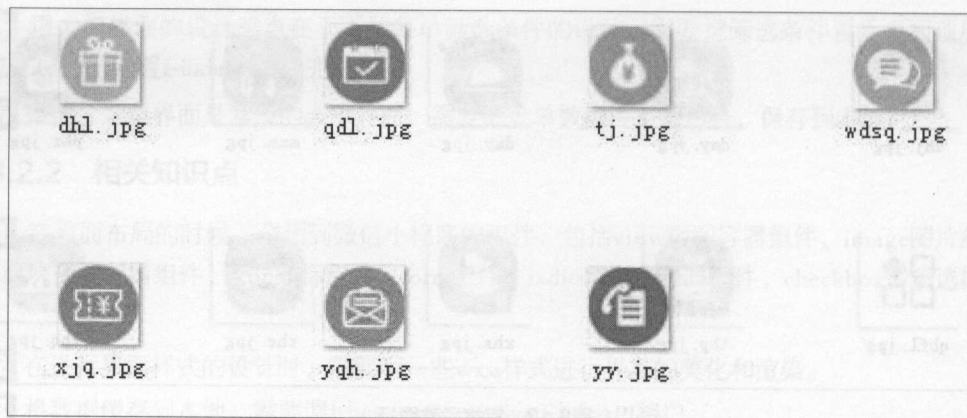


图8.15 我的婚博会图标

8 在现金券界面里，需要用到的一些图标，它们被放置在“images/cash”文件夹下，如图8.16所示。

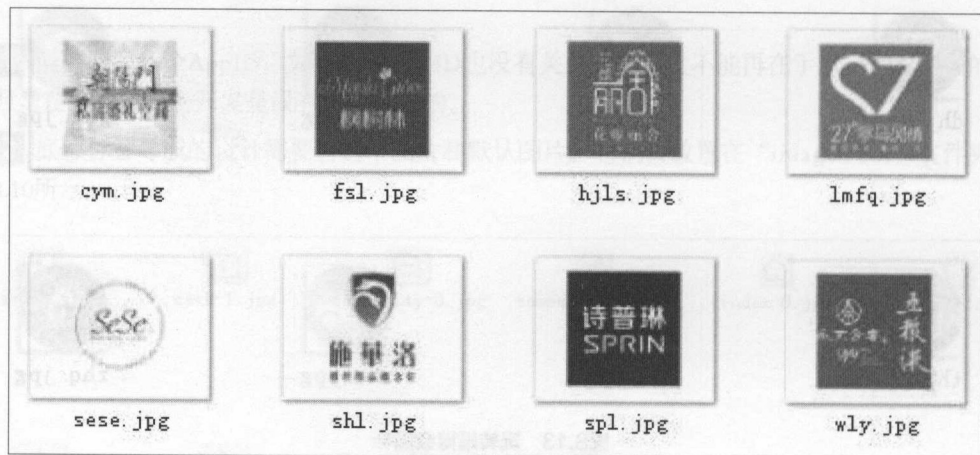


图8.16 现金券图标

9 在婚博会索票界面里，需要用到的一些图片，它们被放置在“images/marry”文件夹下。

8.4 设计流程

仿中国婚博会微信小程序先来设计底部标签导航、海报轮播效果、宫格导航，然后设计全部分类导航界面，再设计现金券下拉菜单筛选条件以及现金券列表，最后设计婚博会索票界面以及获知渠道弹出层。

8.4.1 底部标签导航设计

仿中国婚博会微信小程序有5个底部标签导航：首页、现金券、婚博会、社区、我的。选中标签导航时导航图标和导航文字都会变为红色，如图8.17所示。



图8.17 底部标签导航选中效果

1 新建一个hbb项目的微信小程序，将准备好的底部标签导航图标、海报轮播图片、宫格导航图标、现金券图标、婚博会索票图标放置在hbb项目下。

2 打开App.json配置文件，在pages数组里添加5个页面路径“pages/index/index”“pages/cash/cash”“pages/marry/marry”“pages/community/community”“pages/me/me”，保存后会自动生成相应的页面文件夹；删除掉“pages/logs/logs”页面路径以及对应的文件夹，具体代码如下。

```
{
  "pages": [
    "pages/index/index",
    "pages/cash/cash",
    "pages/marry/marry",
    "pages/community/community",
    "pages/me/me"
  ],
  "window": {
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#fff",
    "navigationBarTitleText": "WeChat",
    "navigationBarTextStyle": "black"
  }
}
```

3 在window数组里配置窗口导航背景颜色为红色（#D73E3E），导航栏文字为“中国婚博会”，字体颜色设置为白色，具体代码如下。

```
{
  "pages": [
    "pages/index/index",
    "pages/cash/cash",
    "pages/marry/marry",
    "pages/community/community",
    "pages/me/me"
  ],
  "window": {
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#D73E3E",
    "navigationBarTitleText": "中国婚博会",
    "navigationBarTextStyle": "white"
  }
}
```


4 在tabBar对象里配置底部标签导航背景色为灰色（# F3F1EF），文字默认颜色为灰色，选中时为红色（#D73E3E），在list数组里配置底部标签导航对应的页面、导航名称、默认时图标、选中时图标，具体代码如下。

```
{
  "pages": [
    "pages/index/index",
    "pages/cash/cash",
    "pages/marry/marry",
    "pages/community/community",
    "pages/me/me"
  ],
  "window": {
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#D73E3E",
    "navigationBarTitleText": "中国婚博会",
    "navigationBarTextStyle": "white"
  },
  "tabBar": {
    "selectedColor": "#D73E3E",
    "backgroundColor": "#F3F1EF",
    "borderStyle": "white",
    "list": [{
      "pagePath": "pages/index/index",
      "text": "首页",
      "iconPath": "images/bar/index-0.jpg",
      "selectedIconPath": "images/bar/index-1.jpg"
    }, {
      "pagePath": "pages/cash/cash",
      "text": "现金券",
      "iconPath": "images/bar/cash-0.jpg",
      "selectedIconPath": "images/bar/cash-1.jpg"
    }, {
      "pagePath": "pages/marry/marry",
      "text": "婚博会",
      "iconPath": "images/bar/marry-0.jpg",
      "selectedIconPath": "images/bar/marry-1.jpg"
    }, {
      "pagePath": "pages/community/community",
      "text": "社区",
      "iconPath": "images/bar/community-0.jpg",
      "selectedIconPath": "images/bar/community-1.jpg"
    }, {
      "pagePath": "pages/me/me",
      "text": "我的",
      "iconPath": "images/bar/me-0.jpg",
      "selectedIconPath": "images/bar/me-1.jpg"
    }
  ]
}
```

这样就完成了仿中国婚博会微信小程序底部标签导航的配置，单击不同的导航，可以切换显示不同的页面，同时导航图标和导航文字会呈现为选中状态。

8.4.2 海报轮播效果设计

海报轮播效果是指在有限的区域内动态地显示不同的广告图片或者商品图片，是很多网站或者App软件都会采用的一种展现方式。在仿中国婚博会微信小程序的首页里，就采用了海报轮播效果展示广告图片，如图8.18所示。



图8.18 海报轮播效果

1 进入到pages/index/index.wxml文件，采用view、swiper、image进行布局，将图片宽度设置为100%，高度设置为80px，具体代码如下。

```
<view class="haibao">
  <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval=
    "{{interval}}" duration="{{duration}}">
    <block wx:for="{{imgUrls}}">
      <swiper-item>
        <image src="{{item}}" class="slide-image" style="width:100%;height:176px;">
      </image>
    </swiper-item>
    </block>
  </swiper>
</view>
```

swiper滑块视图容器设置为自动播放（autoplay=“true”），自动切换时间间隔为3s（interval=“3000”），滑动动画时长为1s（duration=“1000”）。

采用wx:for循环来显示要展示的图片，从index.js里获取imgUrls图片路径。

2 进入到pages/index/index.js文件，在data对象里定义imgUrls数组，存放海报轮播的图片路径，具体代码如下。

```
Page({
  data: {
    indicatorDots: false,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [
      "../../images/haibao/1.jpg",
      "../../images/haibao/2.jpg",
      "../../images/haibao/3.jpg",
      "../../images/haibao/4.jpg"
    ]
  },
  onLoad: function(options) {
    // 页面初始化 options 为页面跳转所带来的参数
  }
})
```

这样就可以实现海报轮播效果，如图8.19和图8.20所示。

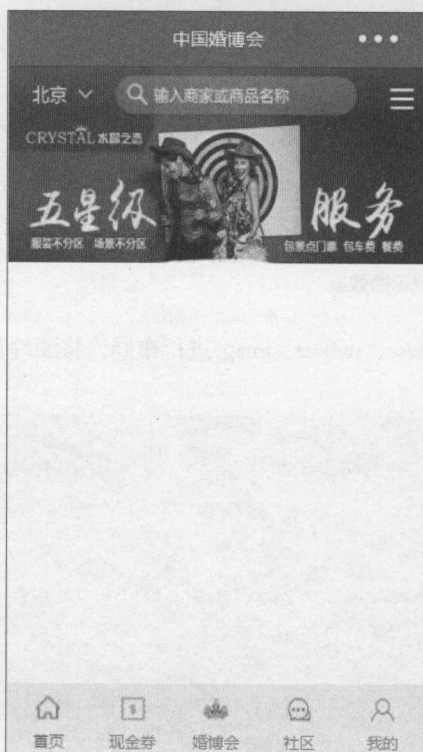


图8.19 海报轮播一

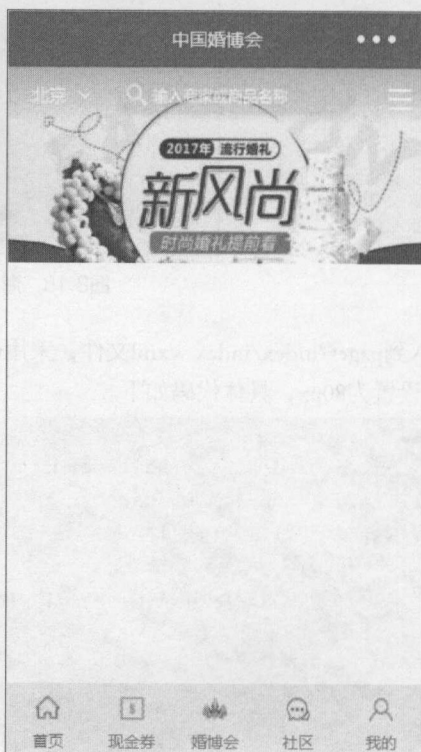


图8.20 海报轮播二

8.4.3 宫格导航设计

宫格导航设计是很多App软件都会采用的一种设计方式。通过宫格导航设计，在首页里就给用户提供明确的入口，用户根据自己的需要通过导航名称进入相关界面，如图8.21所示。



图8.21 宫格导航设计

1 进入pages/index/index.js文件，先准备好要显示的图标和导航名称的数据值，然后定义变量navs，将数据值赋值给navs变量，具体代码如下。

```
Page({
  data: {
    indicatorDots: false,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrl: [
      ".../images/haibao/1.jpg",
      ".../images/haibao/2.jpg",
      ".../images/haibao/3.jpg",
      ".../images/haibao/4.jpg"
    ],
    navs: []
  },
})
```

```
onLoad: function (options) {
    var page = this;
    var navs = this.loadNavData();
    page.setData({ navs: navs });
},
loadNavData: function () {
    var navs = [];
    var nav0 = new Object();
    nav0.img = '../..images/nav/dxy.jpg';
    nav0.name = '订喜宴';
    navs[0] = nav0;

    var nav1 = new Object();
    nav1.img = '../..images/nav/phz.jpg';
    nav1.name = '拍婚照';
    navs[1] = nav1;

    var nav2 = new Object();
    nav2.img = '../..images/nav/zhq.jpg';
    nav2.name = '找婚庆';
    navs[2] = nav2;

    var nav3 = new Object();
    nav3.img = '../..images/nav/dhj.jpg';
    nav3.name = '订婚戒';
    navs[3] = nav3;

    var nav4 = new Object();
    nav4.img = '../..images/nav/xhs.jpg';
    nav4.name = '选婚纱';
    navs[4] = nav4;

    var nav5 = new Object();
    nav5.img = '../..images/nav/thp.jpg';
    nav5.name = '淘婚品';
    navs[5] = nav5;

    var nav6 = new Object();
    nav6.img = '../..images/nav/dmy.jpg';
    nav6.name = '度蜜月';
    navs[6] = nav6;

    var nav7 = new Object();
    nav7.img = '../..images/nav/zhc.jpg';
    nav7.name = '租婚车';
    navs[7] = nav7;

    var nav8 = new Object();
    nav8.img = '../..images/nav/mxn.jpg';
    nav8.name = '美新娘';
```

```

    navs[8] = nav8;

    var nav9 = new Object();
    nav9.img = '../images/nav/qbfl.jpg';
    nav9.name = '全部分类';
    navs[9] = nav9;
    return navs;
  }
})

```

2 进入pages/index/index.wxml文件，进行宫格导航的布局，采用wx:for列表渲染的方式将变量navs值循环显示出来，定义绑定事件navBtn，具体代码如下。

```

<view class="haibao">
  <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval=
    "{{interval}}" duration="{{duration}}">
    <block wx:for="{{imgUrls}}">
      <swiper-item>
        <image src="{{item}}" class="silde-image" style="width:100%;height:176px;">
      </image>
    </swiper-item>
  </block>
</swiper>
</view>
<view class="nav">
  <block wx:for="{{navs}}">
    <view class="item" bindtap="navBtn" id="{{index}}">
      <view>
        <image src="{{item.img}}" style="width:58px;height:56px;"></image>
      </view>
      <view>
        {{item.name}}
      </view>
    </view>
  </block>
</view>
<view class="hr"></view>

```

3 进入pages/index/index.wxss文件，给宫格导航和间隔线添加相应的样式，具体代码如下。

```

.nav{
  text-align: center;
}
.item{
  margin-top:15px;
  text-align: center;
  font-family: "Microsoft YaHei";
  font-size: 13px;
  width: 60px;
  display: inline-block;
  margin-right:10px;
}
.hr{

```



```

height: 1px;
background-color: #cccccc;
opacity: 0.2;
margin-top: 10px;
}

```

4 在App.json新配置一个全部分类type页面文件路径，再进入pages/index/index.js文件，添加navBtn事件用于单击全部分类官格导航时，跳转到全部分类界面，具体代码如下。

```

Page({
  data: {
    indicatorDots: false,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [
      "../../images/haibao/1.jpg",
      "../../images/haibao/2.jpg",
      "../../images/haibao/3.jpg",
      "../../images/haibao/4.jpg"
    ],
    navs: []
  },
  onLoad: function (options) {
    var page = this;
    var navs = this.loadNavData();
    page.setData({ navs: navs });
  },
  navBtn: function (e) {
    console.log(e);
    var id = e.currentTarget.id;
    if (id == "9") {
      wx.navigateTo({
        url: '../type/type'
      })
    }
  },
  loadNavData: function () {
    var navs = [];
    var nav0 = new Object();
    nav0.img = '../../images/nav/dxy.jpg';
    nav0.name = '订喜宴';
    navs[0] = nav0;

    var nav1 = new Object();
    nav1.img = '../../images/nav/phz.jpg';
    nav1.name = '拍婚照';
    navs[1] = nav1;

    var nav2 = new Object();
    nav2.img = '../../images/nav/zhq.jpg';

```

```
nav2.name = '找婚庆';
navs[2] = nav2;

var nav3 = new Object();
nav3.img = '../../images/nav/dhj.jpg';
nav3.name = '订婚戒';
navs[3] = nav3;

var nav4 = new Object();
nav4.img = '../../images/nav/xhs.jpg';
nav4.name = '选婚纱';
navs[4] = nav4;

var nav5 = new Object();
nav5.img = '../../images/nav/thp.jpg';
nav5.name = '淘婚品';
navs[5] = nav5;

var nav6 = new Object();
nav6.img = '../../images/nav/dmy.jpg';
nav6.name = '度蜜月';
navs[6] = nav6;

var nav7 = new Object();
nav7.img = '../../images/nav/zhc.jpg';
nav7.name = '租婚车';
navs[7] = nav7;

var nav8 = new Object();
nav8.img = '../../images/nav/mxn.jpg';
nav8.name = '美新娘';
navs[8] = nav8;

var nav9 = new Object();
nav9.img = '../../images/nav/qbfl.jpg';
nav9.name = '全部分类';
navs[9] = nav9;
return navs;
}
})
```

这样就可以实现宫格导航功能，同时单击全部分类导航时会跳转到全部分类界面。如果想单击其他宫格导航进行跳转，也需要配置相应的跳转界面。

8.4.4 全部分类导航设计

全部分类导航界面集合了中国婚博会所有的导航入口，分为3类：玩转婚博会导航，有订婚宴、拍婚照、找婚庆等服务；特色分类导航，有拍写真、美新娘等服务；我的婚博会导航，有现金券、邀请函、签到礼等服务。这3类导航的布局方式一样，可以先设计一类导航，然后直接进行复用，如图8.22所示。



图8.22 全部分类界面

1 进入到pages/index/type.wxml文件，先来设计玩转婚博会类导航，具体代码如下。

```
<view class="content">
<view class="line"></view>
<view class="item">
  <view class="title">玩转婚博会</view>
  <view class="hr"></view>
  <view class="navs">
    <view class="nav">
      <view><image src="../../images/type/wzhbh/dxy.jpg" style="width:38px;height:
38px;"></image></view>
      <view>订喜宴</view>
    </view>
    <view class="nav">
      <view><image src="../../images/type/wzhbh/phz.jpg" style="width:38px;height:
38px;"></image></view>
      <view>拍婚纱照</view>
    </view>
    <view class="nav">
      <view><image src="../../images/type/wzhbh/zhq.jpg" style="width:38px;height:
38px;"></image></view>
      <view>找婚庆</view>
    </view>
  </view>
</view>
```



```

    </view>
    <view class="nav">
      <view><image src="../../images/type/wzhbh/dhj.jpg" style="width:38px;height:
38px;"></image></view>
      <view>订婚戒</view>
    </view>
  </view>
  <view class="navs">
    <view class="nav">
      <view><image src="../../images/type/wzhbh/xhs.jpg" style="width:38px;height:
38px;"></image></view>
      <view>选婚纱</view>
    </view>
    <view class="nav">
      <view><image src="../../images/type/wzhbh/thp.jpg" style="width:38px;height:
38px;"></image></view>
      <view>淘婚品</view>
    </view>
    <view class="nav">
      <view><image src="../../images/type/wzhbh/dmy.jpg" style="width:38px;height:
38px;"></image></view>
      <view>度蜜月</view>
    </view>
    <view class="nav">
      <view><image src="../../images/type/wzhbh/zhc.jpg" style="width:38px;height:
38px;"></image></view>
      <view>租婚车</view>
    </view>
  </view>
</view>
</view>

```

2 进入pages/index/type.wxss文件，给玩转婚博会类导航添加样式，让它以2行4列的方式展现出来，具体代码如下。

```

.content{
  font-family: "Microsoft YaHei";
  background-color: #F0F0F0;
}
.line{
  height: 10px;
}
.item{
  border: 1px solid #cccccc;
  width: 90%;
  margin: 0 auto;
  background-color: #ffffff;
  padding: 10px;
  border-radius: 5px;
}
.hr{

```

```
height: 1px;
background-color: #cccccc;
opacity: 0.2;
margin-top: 10px;
margin-bottom: 10px;
}
.navs{
  display: flex;
  flex-direction: row;
  text-align: center;
  font-size: 13px;
  margin-bottom: 10px;
  padding-top: 10px;
}
.nav{
  margin: 0 auto;
  width: 70px;
}
```

界面效果如图8.23所示。



图8.23 玩转婚博会导航

3 玩转婚博会导航设计完之后，可以直接复制玩转婚博会导航的内容进行特色分类导航的设计，然后再在这个基础上进行修改，进入pages/index/type.wxml文件，复制玩转婚博会区域，并将其修改成特色分类导航的内容，具体代码如下。

```
<view class="content">
<view class="line"></view>
<view class="item">
  <view class="title"> 玩转婚博会 </view>
  <view class="hr"></view>
  <view class="navs">
    <view class="nav">
      <view><image src="../../images/type/wzhbh/dxy.jpg" style="width:38px;height:
38px;"></image></view>
      <view> 订婚宴 </view>
    </view>
```

```

<view class="nav">
  <view><image src="../../images/type/wzhbh/phz.jpg" style="width:38px;height:
38px;"></image></view>
  <view> 拍婚照 </view>
</view>
<view class="nav">
  <view><image src="../../images/type/wzhbh/zhq.jpg" style="width:38px;height:
38px;"></image></view>
  <view> 找婚庆 </view>
</view>
<view class="nav">
  <view><image src="../../images/type/wzhbh/dhj.jpg" style="width:38px;height:
38px;"></image></view>
  <view> 订婚戒 </view>
</view>
</view>
<view class="navs">
  <view class="nav">
    <view><image src="../../images/type/wzhbh/xhs.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 选婚纱 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wzhbh/thp.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 淘婚品 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wzhbh/dmy.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 度蜜月 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wzhbh/zhc.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 租婚车 </view>
  </view>
</view>
</view>

<view class="line"></view>
<view class="item">
  <view class="title"> 特色分类 </view>
  <view class="hr"></view>
  <view class="navs">
    <view class="nav">
      <view><image src="../../images/type/tsfl/pxz.jpg" style="width:38px;height:38
px;"></image></view>
      <view> 拍写真 </view>
    </view>
  </view>

```



```

<view class="nav">
  <view><image src="../../images/type/tsfl/mxn.jpg" style="width:38px;height:38px;"></image></view>
  <view> 美新娘 </view>
</view>
<view class="nav">
  <view><image src="../../images/type/tsfl/zxj.jpg" style="width:38px;height:38px;"></image></view>
  <view> 装新家 </view>
</view>
<view class="nav">
  <view><image src="../../images/type/tsfl/yyt.jpg" style="width:38px;height:38px;"></image></view>
  <view> 孕婴童 </view>
</view>
</view>
</view>
</view>

```

界面效果如图8.24所示。



图8.24 特色分类导航

4 我的婚博会导航区域也是这样设计的。进入pages/index/type.wxml文件，复制玩转婚博会区域，并将其修改成我的婚博会导航的内容，具体代码如下。

```

<view class="content">
<view class="line"></view>
<view class="item">
  <view class="title">玩转婚博会</view>
  <view class="hr"></view>
  <view class="navs">
    <view class="nav">
      <view><image src="../../images/type/wzhbh/dxy.jpg" style="width:38px;height:

```

```

38px;"></image></view>
    <view> 订喜宴 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wzhbh/phz.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 拍婚照 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wzhbh/zhq.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 找婚庆 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wzhbh/dhj.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 订婚戒 </view>
  </view>
</view>
<view class="navs">
  <view class="nav">
    <view><image src="../../images/type/wzhbh/xhs.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 选婚纱 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wzhbh/thp.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 淘婚品 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wzhbh/dmy.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 度蜜月 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wzhbh/zhc.jpg" style="width:38px;height:
38px;"></image></view>
    <view> 租婚车 </view>
  </view>
</view>

<view class="line"></view>
<view class="item">
  <view class="title"> 特色分类 </view>
  <view class="hr"></view>
  <view class="navs">
    <view class="nav">
      <view><image src="../../images/type/tsfl/pxz.jpg" style="width:38px;height:38

```

```
px;"></image></view>
    <view> 拍写真 </view>
</view>
<view class="nav">
    <view><image src="../../images/type/tsfl/mxn.jpg" style="width:38px;height:38
px;"></image></view>
    <view> 美新娘 </view>
</view>
<view class="nav">
    <view><image src="../../images/type/tsfl/zxj.jpg" style="width:38px;height:38
px;"></image></view>
    <view> 装新家 </view>
</view>
<view class="nav">
    <view><image src="../../images/type/tsfl/yvt.jpg" style="width:38px;height:38
px;"></image></view>
    <view> 孕婴童 </view>
</view>
</view>
</view>

<view class="line"></view>
<view class="item">
    <view class="title"> 我的婚博会 </view>
    <view class="hr"></view>
    <view class="navs">
        <view class="nav">
            <view><image src="../../images/type/wdhbh/xjq.jpg" style="width:38px;height:
38px;"></image></view>
            <view> 现金券 </view>
        </view>
        <view class="nav">
            <view><image src="../../images/type/wdhbh/yqh.jpg" style="width:38px;height:
38px;"></image></view>
            <view> 邀请函 </view>
        </view>
        <view class="nav">
            <view><image src="../../images/type/wdhbh/qdl.jpg" style="width:38px;height:
38px;"></image></view>
            <view> 签到礼 </view>
        </view>
        <view class="nav">
            <view><image src="../../images/type/wdhbh/dhl.jpg" style="width:38px;height:
38px;"></image></view>
            <view> 兑好礼 </view>
        </view>
    </view>
    <view class="navs">
        <view class="nav">
            <view><image src="../../images/type/wdhbh/wdsq.jpg" style="width:38px;height:
```



```

38px;"></image></view>
    <view> 我的社区 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wdhhb/yy.jpg" style="width:38px;height:38
px;"></image></view>
    <view> 预约到婚博会 </view>
  </view>
  <view class="nav">
    <view><image src="../../images/type/wdhhb/tj.jpg" style="width:38px;height:38
px;"></image></view>
    <view> 推荐好友送现金 </view>
  </view>
  <view class="nav">
    <view>
  </view>
</view>
<view class="line"></view>
</view>

```

界面效果如图8.25所示。



图8.25 我的婚博会导航

5 进入pages/type/type.json文件，修改窗口标题，具体代码如下。

```
{
  "navigationBarTitleText": " 全部分类 "
}
```

微信小程序在开发过程中总会遇到一样布局的区域或者类似布局的区域，这时就可以新设计一个区域。如果能将其提炼成模板，则可以提炼成模板使用；如果不能，则可以直接复制这个区域，并在这个基础上进行修改，以提高开发效率。

8.4.5 现金券下拉菜单筛选条件设计

在很多App里都会采用下拉菜单作为筛选条件，下拉菜单里会展现出很多筛选条件供用户选择，现金券界面也是采用下拉菜单来进行条件筛选的。其下拉菜单的筛选条件分为两类：全部和默认，如图8.26和图8.27所示。

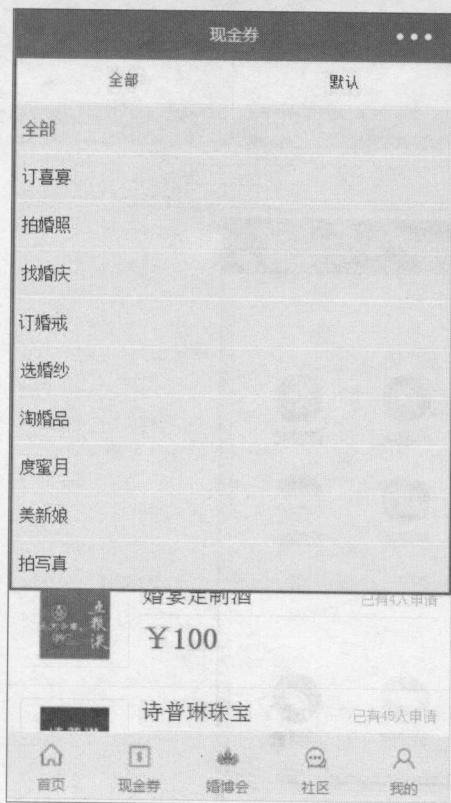


图8.26 全部筛选条件



图8.27 默认筛选条件

1 进入pages/index/cash.wxml文件，进行筛选条件的布局，使用dl、dt、dd进行布局，绑定菜单切换单击事件tapMainMenu，具体代码如下。

```
<dl class="menu">
  <dt data-index="0" bindtap="tapMainMenu" class="{{currentTab==0?'select':'default'}}">全部</dt>
  <dd class="{{subMenuDispaly[0]}}">
    <ul>
```

```

        <li class="select">全部 </li>
        <li>订婚宴 </li>
        <li>拍婚纱照 </li>
        <li>找婚庆 </li>
        <li>订婚戒 </li>
        <li>选婚纱 </li>
        <li>淘婚品 </li>
        <li>度蜜月 </li>
        <li>美新娘 </li>
        <li>拍写真 </li>
    </ul>
</dd>
<dt data-index="1" bindtap="tapMainMenu" class="{{currentTab==1?'select':'default'}}">默认 </dt>
<dd class="{{subMenuDispaly[1]}}">
    <ul>
        <li class="select">默认 </li>
        <li>最新 </li>
        <li>最热 </li>
    </ul>
</dd>
</dl>

```

2 进入pages/index/cash.wxss文件，给下拉菜单筛选条件添加样式，具体代码如下。

```

.menu{
    display: block;
    height: 38px;
}
.menu dt{
    font-size: 13px;
    float: left;
    width: 49.7%;
    height: 38px;
    border-right: 1px solid #f2f2f2;
    border-bottom: 1px solid #f2f2f2;
    text-align: center;
    line-height: 38px;
    background-color: #ffffff;
}
.menu dd{
    position: absolute;
    width: 100%;
    top: 40px;
    left: 0;
    z-index: 999;
}
.menu li{
    font-size: 14px;
    background-color: #F3F1EF;
    line-height: 40px;
    display: block;

```



```
padding-left: 8px;
border-bottom: 1px solid #ffffff;
}
.select{
  color:red;
}
.show{
  display: block;
}
.hidden{
  display: none;
}
```

界面效果如图8.28所示。

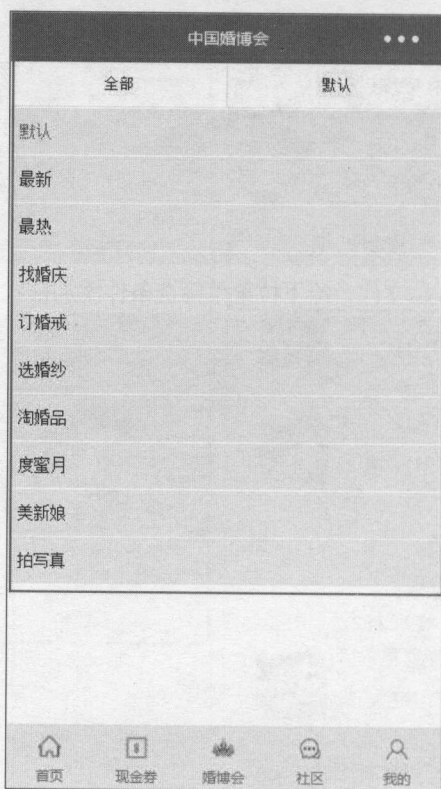


图8.28 筛选条件

3 进入pages/index/cash.js文件，定义两个变量：subMenuDispaly用来控制两个筛选条件内容的显示与隐藏，currentTab用来控制筛选菜单的选中效果，并添加菜单切换事件tapMainMenu，具体代码如下。

```
function initSubMenuDisplay(){
  return ['hidden','hidden'];
}
Page({
  data:{
    subMenuDispaly:initSubMenuDisplay(),
```

```
currentTab:-1
},
tapMainMenu:function(e){
  console.log(e);
  var index = parseInt(e.currentTarget.dataset.index);
  console.log(index);
  var newSubMenuDisplay = initSubMenuDisplay();
  if(this.data.subMenuDispaly[index] == 'hidden'){
    newSubMenuDisplay[index] = 'show';
    this.setData({currentTab:index});
  }else{
    newSubMenuDisplay[index] = 'hidden';
    this.setData({currentTab:-1});
  }
  this.setData({subMenuDispaly:newSubMenuDisplay});
}
})
```

这样就可以实现下拉菜单筛选条件的设计。单击筛选条件菜单名称，菜单名称会呈现为红色选中效果，同时显示出下拉筛选条件选项。单击另一个筛选菜单时，会切换显示，以达到一种动态效果。

8.4.6 现金券列表页设计

现金券列表页用来展示商家的现金优惠券，列表里包括商家图标、商家名称、现金优惠额度以及申请情况，如图8.29所示。



图8.29 现金优惠券列表

1 进入pages/index/cash.wxml文件，先设计一条商家的现金优惠券信息，包括商家图片、商家名称、现金、申请情况，具体代码如下。

```
<dl class="menu">
  <dt data-index="0" bindtap="tapMainMenu" class="{{currentTab==0?'select':'default'}}">全部</dt>
  <dd class="{{subMenuDisplay[0]}}">
    <ul>
      <li class="select">全部</li>
      <li>订喜宴</li>
      <li>拍婚纱照</li>
      <li>找婚庆</li>
      <li>订婚戒</li>
      <li>选婚纱</li>
      <li>淘婚品</li>
      <li>度蜜月</li>
      <li>美新娘</li>
      <li>拍写真</li>
    </ul>
  </dd>
  <dt data-index="1" bindtap="tapMainMenu" class="{{currentTab==1?'select':'default'}}">默认</dt>
  <dd class="{{subMenuDisplay[1]}}">
    <ul>
      <li class="select">默认</li>
      <li>最新</li>
      <li>最热</li>
    </ul>
  </dd>
</dl>
<view class="items">
  <view class="item">
    <view><image src="../../../images/cash/shl.jpg" style="width:93px;height:73px;">
</image></view>
    <view class="des">
      <view class="title">
        施华洛婚纱摄影 <text class="Apply">已有 4 人申请</text>
      </view>
      <view class="hr"></view>
      <view class="price">¥200-300</view>
    </view>
  </view>
  <view class="line"></view>
</view>
```

2 进入pages/index/cash.wxss文件，给现金优惠券信息添加样式，分为左右两列，左侧是商家图片，右侧是现金优惠券相关信息，具体代码如下。

```
.menu{
  display: block;
  height: 38px;
}
```



```
.menu dt{
    font-size: 13px;
    float: left;
    width: 49.7%;
    height: 38px;
    border-right: 1px solid #f2f2f2;
    border-bottom: 1px solid #f2f2f2;
    text-align: center;
    line-height: 38px;
    background-color: #ffffff;
}
.menu dd{
    position: absolute;
    width: 100%;
    top: 40px;
    left: 0;
    z-index: 999;
}
.menu li{
    font-size: 14px;
    background-color: #F3F1EF;
    line-height: 40px;
    display: block;
    padding-left: 8px;
    border-bottom: 1px solid #ffffff;
}
.select{
    color: red;
}
.show{
    display: block;
}
.hidden{
    display: none;
}
.item{
    margin: 10px;
    display: flex;
    flex-direction: row;
}
.des{
    margin-left: 10px;
    width: 100%;
}
.Apply{
    font-size: 12px;
    color: #cccccc;
    position: absolute;
    right: 10px;
    margin-top: 5px;
}
```

```

}
.hr{
  height: 1px;
  background-color: #f2f2f2;
  margin-top: 5px;
  margin-bottom: 5px;
}
.price{
  font-size: 25px;
  color: red;
  font-weight: bold;
}
.line{
  height: 1px;
  background-color: #f2f2f2;
  margin-top: 10px;
  margin-bottom: 10px;
}

```

界面效果如图8.30所示。

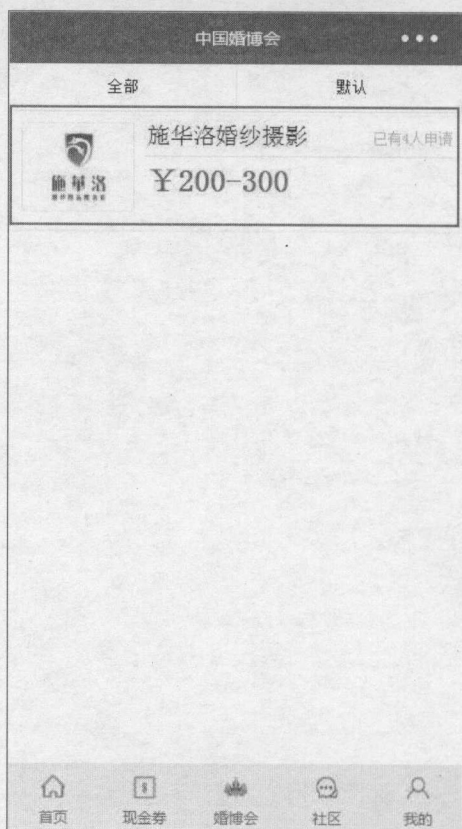


图8.30 现金优惠券的布局

3 进入pages/index/cash.wxml文件，复制设计好的现金优惠券信息，并在此基础上进行修改，具体代码如下。

```

<dl class="menu">
  <dt data-index="0" bindtap="tapMainMenu" class="{{currentTab==0?'select':'default'}}">全部</dt>
  <dd class="{{subMenuDispaly[0]}}">
    <ul>
      <li class="select">全部</li>
      <li>订喜宴</li>
      <li>拍婚照</li>
      <li>找婚庆</li>
      <li>订婚戒</li>
      <li>选婚纱</li>
      <li>淘婚品</li>
      <li>度蜜月</li>
      <li>美新娘</li>
      <li>拍写真</li>
    </ul>
  </dd>
  <dt data-index="1" bindtap="tapMainMenu" class="{{currentTab==1?'select':'default'}}">默认</dt>
  <dd class="{{subMenuDispaly[1]}}">
    <ul>
      <li class="select">默认</li>
      <li>最新</li>
      <li>最热</li>
    </ul>
  </dd>
</dl>
<view class="items">
  <view class="item">
    <view><image src="../../images/cash/shl.jpg" style="width:93px;height:73px;">
</image></view>
    <view class="des">
      <view class="title">
        施华洛婚纱摄影<text class="Apply">已有 4 人申请</text>
      </view>
      <view class="hr"></view>
      <view class="price">¥200-300</view>
    </view>
  </view>
  <view class="line"></view>

  <view class="item">
    <view><image src="../../images/cash/fsl.jpg" style="width:93px;height:73px;">
</image></view>
    <view class="des">
      <view class="title">
        枫树林高端婚礼<text class="Apply">已有 14 人申请</text>
      </view>
      <view class="hr"></view>
      <view class="price">¥100</view>
    </view>
  </view>
</view>

```



```

    </view>
  </view>
  <view class="line"></view>

  <view class="item">
    <view><image src="../../../images/cash/hjls.jpg" style="width:93px;height:73px;">
</image></view>
    <view class="des">
      <view class="title">
        花嫁丽舍一站式 <text class="Apply"> 已有 129 人申请 </text>
      </view>
      <view class="hr"></view>
      <view class="price"> ¥1000</view>
    </view>
  </view>
  <view class="line"></view>

  <view class="item">
    <view><image src="../../../images/cash/sese.jpg" style="width:93px;height:73px;">
</image></view>
    <view class="des">
      <view class="title">
        SeSe 婚礼国王 <text class="Apply"> 已有 117 人申请 </text>
      </view>
      <view class="hr"></view>
      <view class="price"> ¥100</view>
    </view>
  </view>
  <view class="line"></view>

  <view class="item">
    <view><image src="../../../images/cash/wly.jpg" style="width:93px;height:73px;">
</image></view>
    <view class="des">
      <view class="title">
        婚宴定制酒 <text class="Apply"> 已有 4 人申请 </text>
      </view>
      <view class="hr"></view>
      <view class="price"> ¥100</view>
    </view>
  </view>
  <view class="line"></view>

  <view class="item">
    <view><image src="../../../images/cash/spl.jpg" style="width:93px;height:73px;">
</image></view>
    <view class="des">
      <view class="title">
        诗普琳珠宝 <text class="Apply"> 已有 45 人申请 </text>
      </view>

```

```

<view class="hr"></view>
<view class="price">¥100</view>
</view>
</view>
<view class="line"></view>

<view class="item">
  <view><image src="../../images/cash/cym.jpg" style="width:93px;height:73px;">
</image></view>
  <view class="des">
    <view class="title">
      朝阳门私属婚礼空间 <text class="Apply">已有 3 人申请</text>
    </view>
    <view class="hr"></view>
    <view class="price">¥800</view>
  </view>
</view>
<view class="line"></view>
</view>

```

界面效果如图8.31所示。



图8.31 现金券界面

4 进入pages/cash/cash.json文件，修改窗口标题，具体代码如下。

```

{
  "navigationBarTitleText": " 现金券 "
}

```

8.4.7 婚博会索票界面设计

婚博会索票界面用来索要中国婚博会门票，只有有了门票才可参加婚博会。索要门票需要填写自己和爱人的相关信息，如图8.32所示。

图8.32 索票表单

1 进入pages/marry/marry.wxml文件，设计婚博会宣传海报和索票按钮的布局，具体代码如下。

```
<view class="content">
  <view>
    <image src="../../images/marry/xuanchuan.jpg" style="width:100%;height:220px;">
  </image>
  </view>
  <view class="ticket">
    <view class="first"><icon type="success"></icon> 我第一次索票 </view>
    <view class="second"><icon type="success"></icon> 我以前索过票 </view>
  </view>
</view>
```

2 进入pages/marry/marry.wxss文件，给索票按钮添加样式，具体代码如下。

```
.ticket{
  display: flex;
  flex-direction: row;
  width: 100%;
  text-align: center;
  margin-bottom: 30px;
```



```

}
.first{
  border:1px solid red;
  width: 45%;
  margin: 0 auto;
  background-color: #D60210;
  height: 35px;
  line-height: 35px;
  font-size: 14px;
  color: #ffffff;
}
.second{
  border:1px solid red;
  width: 45%;
  margin: 0 auto;
  background-color: #FDA6AE;
  height: 35px;
  line-height: 35px;
  font-size: 14px;
}

```

界面效果如图8.33所示。



图8.33 索票按钮

3 进入pages/marry/marry.wxml文件，设计索票需要填写的表单，需要用到input文本框组件、checkbox多项选择器组件、button按钮组件、radio单项选择器组件、form组件，具体代码如下。

```

<view class="content">
  <view>
    <image src="../../../images/marry/xuanchuan.jpg" style="width:100%;height:220px;">
  </image>
  </view>
  <view class="ticket">
    <view class="first"><icon type="success"></icon> 我第一次索票 </view>
    <view class="second"><icon type="success"></icon> 我以前索过票 </view>
  </view>
</form bindsubmit="formSubmit">

```

```

<view class="item">
  <view class="name"> 姓名: </view>
  <view class="val"><input name="name" type="text"/></view>
</view>
<view class="item">
  <view class="name"> 手机号: </view>
  <view class="val"><input name="mobile" type="text"/></view>
</view>
<view class="item">
  <view class="name"> 爱人姓名: </view>
  <view class="val"><input name="lovename" type="text"/></view>
</view>
<view class="item">
  <view class="name"> 爱人手机: </view>
  <view class="val"><input name="lovemobile" type="text"/></view>
</view>
<view class="item">
  <view class="name"> 快递地址: </view>
  <view class="val"><input name="address" type="text" placeholder=" 情侣套票免费快
递到家 " placeholder-class="holder"/></view>
</view>
<view class="item">
  <view class="name"> 婚期: </view>
  <view class="val"><input name="date" type="text" placeholder=" 请输入日期 "
placeholder-class="holder"/></view>
</view>
<view class="item">
  <view class="name"> 需要筹备: </view>
  <view class="box">
    <checkbox-group name="box">
      <checkbox value=" 订喜宴 "> 订喜宴 </checkbox>
      <checkbox value=" 找婚庆 "> 找婚庆 </checkbox>
      <checkbox value=" 拍婚照 "> 拍婚照 </checkbox>
      <checkbox value=" 订婚戒 "> 订婚戒 </checkbox>
      <checkbox value=" 选婚纱 "> 选婚纱 </checkbox>
      <checkbox value=" 度蜜月 "> 度蜜月 </checkbox>
      <checkbox value=" 淘婚品 "> 淘婚品 </checkbox>
      <checkbox value=" 美新娘 "> 美新娘 </checkbox>
      <checkbox value=" 租婚车 "> 租婚车 </checkbox>
    </checkbox-group>
  </view>
</view>
<view class="item">
  <view class="name"> 获知渠道: </view>
  <view><button class="way" bindtap="selectWay">{{way}}</button></view>
</view>
<button class="btn" form-type="submit"> 免费索票 </button>
</form>
</view>

```

4 进入到pages/marry/marry.wxss文件，给表单添加样式，具体代码如下。

```
.ticket{
  display: flex;
  flex-direction: row;
  width: 100%;
  text-align: center;
  margin-bottom: 30px;
}

.first{
  border:1px solid red;
  width: 45%;
  margin: 0 auto;
  background-color: #D60210;
  height: 35px;
  line-height: 35px;
  font-size: 14px;
  color: #ffffff;
}

.second{
  border:1px solid red;
  width: 45%;
  margin: 0 auto;
  background-color: #FDA6AE;
  height: 35px;
  line-height: 35px;
  font-size: 14px;
}

.item{
  margin: 10px;
  display: flex;
  flex-direction: row;
}

.name{
  width: 100px;
  text-align: right;
  height: 28px;
  line-height: 28px;
  font-size: 15px;
}

.val{
  width: 230px;
  height: 28px;
  border:1px solid #cccccc;
  border-radius: 5px;
  font-size:13px;
}

.holder{
  font-size: 13px;
  margin-left: 5px;
  color: #999999;
}
```



```
.box{
    font-size: 13px;
    width: 300px;
    margin-left: 10px;
}
.box checkbox{
    margin-right: 10px;
    margin-top: 10px;
}
.btn{
    width:220px;
    height: 45px;
    line-height: 45px;
    background-color: #D50310;
    color: #ffffff;
    border-radius: 50px;
}
.way{
    width: 230px;
    height: 30px;
    font-size: 13px;
    text-align: left;
}
```

界面效果如图8.34所示。

中国婚博会

我第一次索票

我以前索过票

姓名：

手机号：

爱人姓名：

爱人手机：

快递地址：

情侣套票免费快速到家

婚期：

请输入日期

需要筹备：

☐

订婚宴

☐

找婚庆

☐

拍婚纱照

☐

订婚戒

☐

选婚纱

☐

度蜜月

☐

淘婚品

☐

美新娘

☐

租婚车

获知渠道：

免费索票

首页

现金券

婚博会

社区

我的

图8.34 索票表单

使用form组件绑定的bindsubmit="formSubmit"事件，表单项添加name属性，然后单击button按钮绑定form-type="submit"属性就可以提交表单，获知渠道的内容可以通过单击获知渠道然后以弹出层的形式展现出来。

8.4.8 获知渠道弹出层设计

获知渠道是以弹出层radio单项选择器组件的列表展现出来的，只能选择一种获知渠道，如图8.35所示。

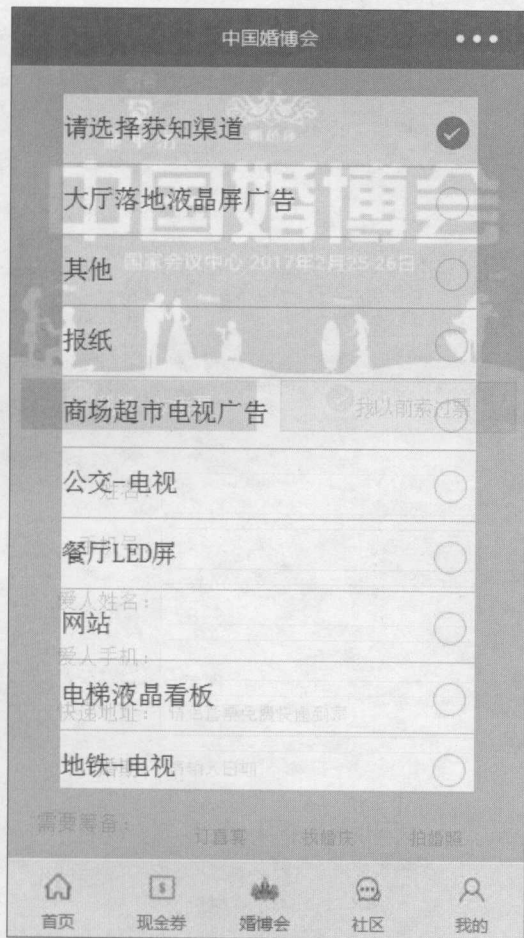


图8.35 获知渠道界面

1 进入pages/marry/marry.wxml文件，设计弹出框布局，并使用radio单项选择器进行列表布局，通过flag变量来控制是否显示弹出层，具体代码如下。

```
<view class="content">
  <view>
    <image src="../../../images/marry/xuanchuan.jpg" style="width:100%;height:220px;"></image>
  </view>
  <view class="ticket">
    <view class="first"><icon type="success"></icon> 我第一次索票 </view>
```

```

    <view class="second"><icon type="success"></icon> 我以前索过票 </view>
</view>

<form bindsubmit="formSubmit">
    <view class="item">
        <view class="name"> 姓名: </view>
        <view class="val"><input name="name" type="text"/></view>
    </view>
    <view class="item">
        <view class="name"> 手机号: </view>
        <view class="val"><input name="mobile" type="text"/></view>
    </view>
    <view class="item">
        <view class="name"> 爱人姓名: </view>
        <view class="val"><input name="lovename" type="text"/></view>
    </view>
    <view class="item">
        <view class="name"> 爱人手机: </view>
        <view class="val"><input name="lovemobile" type="text"/></view>
    </view>
    <view class="item">
        <view class="name"> 快递地址: </view>
        <view class="val"><input name="address" type="text" placeholder=" 情侣套票免费快
递到家 " placeholder-class="holder"/></view>
    </view>
    <view class="item">
        <view class="name"> 婚期: </view>
        <view class="val"><input name="date" type="text" placeholder=" 请输入日期 "
placeholder-class="holder"/></view>
    </view>
    <view class="item">
        <view class="name"> 需要筹备: </view>
        <view class="box">
            <checkbox-group name="box">
                <checkbox value=" 订喜宴 "> 订喜宴 </checkbox>
                <checkbox value=" 找婚庆 "> 找婚庆 </checkbox>
                <checkbox value=" 拍婚照 "> 拍婚照 </checkbox>
                <checkbox value=" 订婚戒 "> 订婚戒 </checkbox>
                <checkbox value=" 选婚纱 "> 选婚纱 </checkbox>
                <checkbox value=" 度蜜月 "> 度蜜月 </checkbox>
                <checkbox value=" 淘婚品 "> 淘婚品 </checkbox>
                <checkbox value=" 美新娘 "> 美新娘 </checkbox>
                <checkbox value=" 租婚车 "> 租婚车 </checkbox>
            </checkbox-group>
        </view>
    </view>
    <view class="item">
        <view class="name"> 获知渠道: </view>
        <view><button class="way" bindtap="selectWay">{{way}}</button></view>
    </view>

```



```

    <button class="btn" form-type="submit">免费索票 </button>
  </form>
</view>
<view class="{{flag=='0'? 'bg': 'hideBg'}}">
  <view class="radioBg">
    <radio-group bindchange="radioChange">
      <view class="radioItem">
        <view class="radioName"> 请选择获知渠道 </view>
        <view class="radioVal"><radio value=" 请选择获知渠道 " checked/></view>
      </view>
      <view class="radioItem">
        <view class="radioName"> 大厅落地液晶屏广告 </view>
        <view class="radioVal"><radio value=" 大厅落地液晶屏广告 " /></view>
      </view>
      <view class="radioItem">
        <view class="radioName"> 其他 </view>
        <view class="radioVal"><radio value=" 其他 " /></view>
      </view>
      <view class="radioItem">
        <view class="radioName"> 报纸 </view>
        <view class="radioVal"><radio value=" 报纸 " /></view>
      </view>
      <view class="radioItem">
        <view class="radioName"> 商场超市电视广告 </view>
        <view class="radioVal"><radio value=" 商场超市电视广告 " /></view>
      </view>
      <view class="radioItem">
        <view class="radioName"> 公交 - 电视 </view>
        <view class="radioVal"><radio value=" 公交 - 电视 " /></view>
      </view>
      <view class="radioItem">
        <view class="radioName"> 餐厅 LED 屏 </view>
        <view class="radioVal"><radio value=" 餐厅 LED 屏 " /></view>
      </view>
      <view class="radioItem">
        <view class="radioName"> 网站 </view>
        <view class="radioVal"><radio value=" 网站 " /></view>
      </view>
      <view class="radioItem">
        <view class="radioName"> 电梯液晶看板 </view>
        <view class="radioVal"><radio value=" 电梯液晶看板 " /></view>
      </view>
      <view class="radioItem">
        <view class="radioName"> 地铁 - 电视 </view>
        <view class="radioVal"><radio value=" 地铁 - 电视 " /></view>
      </view>
    </radio-group>
  </view>
</view>

```

2 进入pages/marry/marry.wxss文件，给弹出框添加样式，置于页面顶层需要使用z-index属性，

具体代码如下。

```
.ticket{
  display: flex;
  flex-direction: row;
  width: 100%;
  text-align: center;
  margin-bottom: 30px;
}
.first{
  border:1px solid red;
  width: 45%;
  margin: 0 auto;
  background-color: #D60210;
  height: 35px;
  line-height: 35px;
  font-size: 14px;
  color: #ffffff;
}
.second{
  border:1px solid red;
  width: 45%;
  margin: 0 auto;
  background-color: #FDA6AE;
  height: 35px;
  line-height: 35px;
  font-size: 14px;
}
.item{
  margin: 10px;
  display: flex;
  flex-direction: row;
}
.name{
  width: 100px;
  text-align: right;
  height: 28px;
  line-height: 28px;
  font-size: 15px;
}
.val{
  width: 230px;
  height: 28px;
  border:1px solid #cccccc;
  border-radius: 5px;
  font-size:13px;
}
.holder{
  font-size: 13px;
  margin-left: 5px;
  color: #999999;
```

```

}
.box{
    font-size: 13px;
    width: 300px;
    margin-left: 10px;
}
.box checkbox{
    margin-right: 10px;
    margin-top: 10px;
}
.btn{
    width:220px;
    height: 45px;
    line-height: 45px;
    background-color: #D50310;
    color: #ffffff;
    border-radius: 50px;
}
.way{
    width: 230px;
    height: 30px;
    font-size: 13px;
    text-align: left;
}
.bg{
    display: block;
    background-color: #cccccc;
    width: 100%;
    height: 750px;
    position: absolute;
    top:0px;
    left:0px;
    z-index: 999;
    opacity: 0.9;
}
.radioBg{
    background-color: #ffffff;
    width: 80%;
    height: 500px;
    margin: 0 auto;
    margin-top:20px;
}
.radioItem{
    display: flex;
    flex-direction: row;
    height: 50px;
    align-items: center;
    border-bottom: 1px solid #cccccc;
}
.radioName{

```



```
width: 90%;  
}  
.hideBg(  
  display: none;  
)
```

3 进入pages/marry/marry.js文件，定义两个变量：flag等于0代表显示弹出框，flag等于1代表不显示弹出框；变量way是获知渠道按钮的名称，添加selectWay绑定事件，单击按钮时，显示弹出框，具体代码如下。

```
Page({  
  data:{  
    flag:'1',  
    way:' 请选择获知渠道 '  
  },  
  selectWay:function(){  
    this.setData({flag:'0'});  
  }  
})
```

4 单击获知渠道时，弹出框显示获知渠道列表，界面效果如图8.36所示。

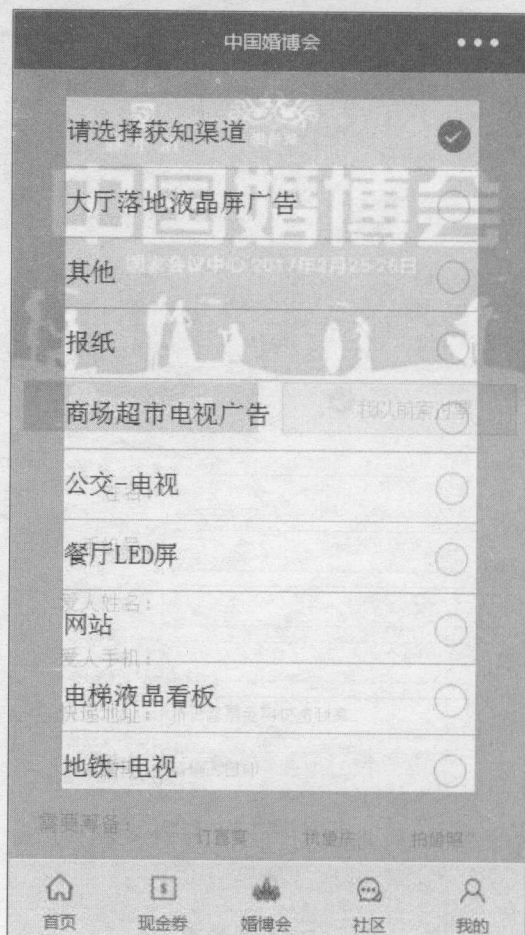


图8.36 获知渠道列表

5 进入pages/marry/marry.js文件，给单项选择器添加radioChange绑定事件，同时设置获知渠道按钮的名称，具体代码如下。

```
Page({
  data: {
    flag: '1',
    way: ' 请选择获知渠道 '
  },
  selectWay: function() {
    this.setData({flag: '0'});
  },
  radioChange: function(e) {
    console.log(e);
    var way = e.detail.value;
    this.setData({flag: '1'});
    this.setData({way: way});
  }
})
```

6 进入pages/marry/marry.js文件，添加form表单绑定formSubmit事件，可以获取表单提交的内容，将表单提交的内容保存到本地，具体代码如下。

```
Page({
  data: {
    flag: '1',
    way: ' 请选择获知渠道 '
  },
  selectWay: function() {
    this.setData({flag: '0'});
  },
  radioChange: function(e) {
    console.log(e);
    var way = e.detail.value;
    this.setData({flag: '1'});
    this.setData({way: way});
  },
  formSubmit: function(e) {
    console.log(e);
    var ticket = e.detail.value;
    ticket.way = this.data.way;
    wx.setStorageSync('ticket', ticket);
  }
})
```

数据提交界面如图8.37所示。

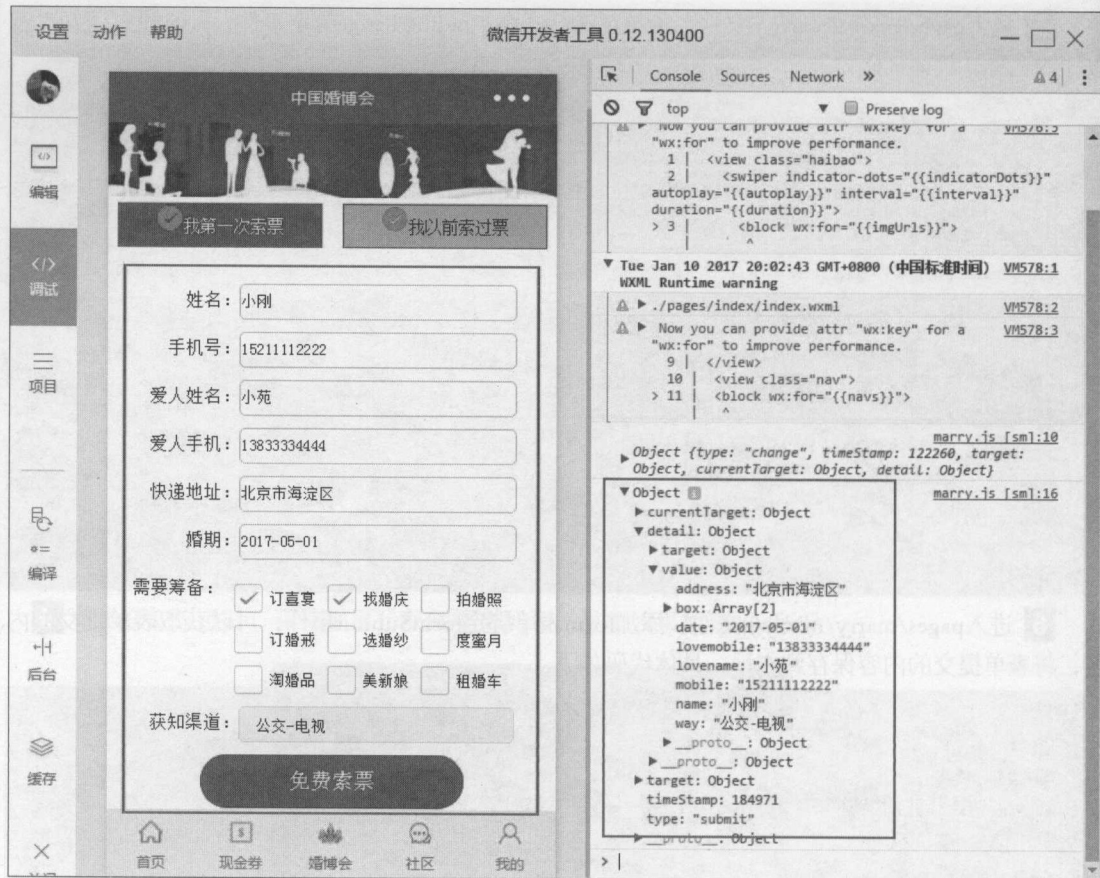


图8.37 数据提交界面

这样就完成了弹出框的设计。将弹出框选项值赋值给按钮，然后将表单数据提交到后台保存起来。

8.5 小结

本章主要设计了仿中国婚博会微信小程序，重点掌握以下内容。

- 1 学会利用微信小程序来完成界面的布局以及给界面添加相关的布局样式。
- 2 学会底部标签导航、海报轮播效果、宫格导航的设计。
- 3 在界面布局时，如果有类似布局或者一样的布局，可以先设计一个布局和样式，然后再复用这个布局和样式，以提高开发效率。
- 4 学会制作下拉菜单单选条件，动态地切换不同的下拉菜单。
- 5 学会表单制作，设计表单样式以及使用表单组件。
- 6 学会弹出窗的设计，动态地控制显示与隐藏效果。



通过本书怎么学会微信小程序开发?

Step1

图文代码快速理解小程序的基本原理和应用方法

先明白学什么，用来做什么

再用图+表+示例代码+详细代码说明，理解基本原理

Step2

沙场大练兵——边做边学

学完马上实战演练

先看案例最终效果，马上就会做

Step3

综合实战，感受真实商业项目的制作过程

完整运用所学知识

商业项目零距离接触



不只学编程，还学调研和设计

本书适合谁看?



所有想成为“小程序员”的人，无论你已经是程序员，
还是设计师、学生、创业青年、网虫、策划人员、编辑

.....

免/费/提/供

PPT等教学相关资料



人邮教育

www.ryjiaoyu.com

教材服务热线: 010-81055256

反馈/投稿/推荐信箱: 315@ptpress.com.cn

人民邮电出版社教育服务与资源下载社区: www.ryjiaoyu.com

ISBN 978-7-115-45045-6



9 787115 450456 >

ISBN 978-7-115-45045-6

定价: 59.80 元